(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

REVISED VERSION

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 November 2001 (22.11.2001)

PCT

(10) International Publication Number
WO 01/88811 A2

(51) International Patent Classification[7]: G06F 17/60

(21) International Application Number: PCT/US01/15424

(22) International Filing Date: 12 May 2001 (12.05.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/203,849 12 May 2000 (12.05.2000) US

(71) Applicant: INVISIBLE HAND NETWORKS, INC. [US/US]; 527 West 34th Street, 6th Floor, New York, NY 10001 (US).

(72) Inventors: SEMRET, Nemo; 123 Avenue A, New York, NY 10009 (US). GIAMMARINO, Giovanna; 123 Avenue A, New York, NY 10009 (US).

(74) Agents: MAJERUS, Laura, A. et al.; Fenwick & West LLP, Two Palo Alto Square, Palo Alto, CA 94306 (US).

(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with declaration under Article 17(2)(a); without abstract; title not checked by the International Searching Authority*

(48) Date of publication of this revised version:
21 February 2002

(15) Information about Correction:
see PCT Gazette No. 08/2002 of 21 February 2002, Section II

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/88811 A2

(54) Title: METHOD AND SYSTEM FOR MARKET BASED RESOURCE ALLOCATION

(57) Abstract:

PATENT COOPERATION TREATY

PCT

DECLARATION OF NON-ESTABLISHMENT OF INTERNATIONAL SEARCH REPORT

(PCT Article 17(2)(a), Rules 13ter.1(c) and Rule 39)

Applicant's or agent's file reference	IMPORTANT DECLARATION	Date of mailing *(day/month/year)*
4980 PCT		05/11/2001
International application No.	International filing date *(day/month/year)*	(Earliest) Priority date *(day/month/year)*
PCT/US 01/ 15424	12/05/2001	12/05/2000
International Patent Classification (IPC) or both national classification and IPC	G06F17/60	
Applicant		
INVISIBLE HAND NETWORKS, INC.		

This International Searching Authority hereby declares, according to Article 17(2)(a), that **no international search report will be established** on the international application for the reasons indicated below

1. [X] The subject matter of the international application relates to:
   - a. [ ] scientific theories.
   - b. [ ] mathematical theories
   - c. [ ] plant varieties.
   - d. [ ] animal varieties.
   - e. [ ] essentially biological processes for the production of plants and animals, other than microbiological processes and the products of such processes.
   - f. [X] schemes, rules or methods of doing business.
   - g. [ ] schemes, rules or methods of performing purely mental acts.
   - h. [ ] schemes, rules or methods of playing games.
   - i. [ ] methods for treatment of the human body by surgery or therapy.
   - j. [ ] methods for treatment of the animal body by surgery or therapy.
   - k. [ ] diagnostic methods practised on the human or animal body.
   - l. [ ] mere presentations of information.
   - m. [ ] computer programs for which this International Searching Authority is not equipped to search prior art.

2. [ ] The failure of the following parts of the international application to comply with prescribed requirements prevents a meaningful search from being carried out:
   - [ ] the description
   - [ ] the claims
   - [ ] the drawings

3. [ ] The failure of the nucleotide and/or amino acid sequence listing to comply with the standard provided for in Annex C of the Administrative Instructions prevents a meaningful search from being carried out:
   - [ ] the written form has not been furnished or does not comply with the standard.
   - [ ] the computer readable form has not been furnished or does not comply with the standard.

4. Further comments:

Name and mailing address of the International Searching Authority	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL-2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Lucia Van Pinxteren

Form PCT/ISA/203 (July 1998)

FURTHER INFORMATION CONTINUED FROM PCT/ISA/ 203

The claims relate to subject matter for which no search is required according to Rule 39 PCT. Given that the claims are formulated in terms of such subject matter or merely specify commonplace features relating to its technological implementation, the search examiner could not establish any technical problem which might potentially have required an inventive step to overcome. Hence it was not possible to carry out a meaningful search into the state of the art (Art. 17(2)(a)(i) and (ii) PCT; see Guidelines Part B Chapter VIII, 1-6).

The applicant's attention is drawn to the fact that claims relating to inventions in respect of which no international search report has been established need not be the subject of an international preliminary examination (Rule 66.1(e) PCT). The applicant is advised that the EPO policy when acting as an International Preliminary Examining Authority is normally not to carry out a preliminary examination on matter which has not been searched. This is the case irrespective of whether or not the claims are amended following receipt of the search report or during any Chapter II procedure. If the application proceeds into the regional phase before the EPO, the applicant is reminded that a search may be carried out during examination before the EPO (see EPO Guideline C-VI, 8.5), should the problems which led to the Article 17(2) declaration be overcome.

In general, the present invention promotes the sharing of a limited resource, such as bandwidth, buffer space, memory space, storage, or processor time, in a competitive environment. This environment ensures that whomever need the most resource and has the ability to pay will get a share of the resource in accordance with willingness to pay. In addition, the invention adjusts for the changing needs of the participants over time.

Advantages of the invention will be set forth in part in the description which follows and in part will be apparent from the description or may be learned by practice of the invention. The objects and advantages of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims and equivalents.

Brief Description of the Drawings

Fig. 1 is a block diagram of an embodiment of the present invention.

Fig. 2 is a flow chart of a method performed by a resource agent of Fig. 1.

Fig. 3 is a flow chart of a method performed by a of Fig. 1.

Fig. 4 is a flow chart of a method performed by a seller agent of Fig. 1.

Fig. 5 is a chart showing an example of the result of an allocation rule.

Figs. 6(a) and 6(b) show an example of a valuation rule.

Fig. 7(a) and 7(b) show an example of a strategy rule.

Fig. 8 shows an example of a flow chart used by an accounting system of an embodiment of the invention.

Fig. 9(a) is a more detailed block diagram of the embodiment of Fig. 1.

Fig. 9(b) shows an example of how a bandwidth resource is given to the winning bidder.

Fig. 10 shows an example of an embodiment of the present invention including a "garage" for player agents.

Figs. 11(a)-11(b) are an example of an XML file for a generic player agent.

Figs. 12(a)-12(c) shown an example of Java code implementing a strategy rule for a

Figs. 13(a)-13(c) show an example of Java code implementing an allocation rule for a resource agent 104.

Fig. 14 is an example of an XML file for a generic resource agent.

Figs. 15(a)-15( ) show examples of user interfaces for buyer and seller agents.

Detailed Description of Embodiments

Reference will now be made in detail to several embodiments of the present invention, examples of which are illustrated in the accompanying drawings. Wherever practicable, the same reference numbers will be used throughout the drawings to refer to the same or like parts.

The present invention uses distributed, self-optimizing software agents to perform resource allocation more efficiently than centralized or human-based systems.

Fig. 1 is a block diagram of an embodiment of the present invention. Fig. 1 includes a software player agent 102, which represents multiple and seller agents. A typical system will, include both buyer and seller agents 102. Fig. 1 also includes a software resource agent 104, a software accounting agent 106, a network control and management agent 108 and a resource 110. It is contemplated that resource 110 can be a number of different resources, including but not limited to: bandwidth, buffer space, memory space, storage, or processor time. As shown in Fig. 1, each player agent (buyer and seller agent) can contain a Graphical User Interface (GUI) 122, one or more valuation rules 124, one or more strategy rules 126, and one or more allocation rules 128. The resource agent 104 also contains the same one or more allocation rules 128.

s 102 place bids to the resource agent 104, which ultimately decides which of the player agents is awarded a portion of each resource for a predetermined amount of time.

Resource agent 104 also controls resource 110, based on the results of the bidding, by sending an allocation command to network control and management agent 108. Thus, for example, if the resource is Internet bandwidth, if buyer A has won X Mbs of bandwidth for 5 minutes, resource agent 104 directs the network control and management agent 108 to give precedence to packets originating at buyer A for the next five minutes or to ensure that the agreed upon bandwidth requirements are met for buyer A's packets for the next five minutes. The mechanism used to communicate the allocation command from resource agent 104 to agent 108 can be any appropriate mechanism. In the example shown, the allocation command will include an identification of the winning buyer or buyers and an identification of the amount of resource allocated and the time period for which it is allocated. Other appropriate formats can be used without departing from the spirit of the invention. In some embodiments, agents 104 and 108 are separate agents (as shown in the figure) in other embodiments, their functions are merged into a single agent. Agents 104

and 108 can be owned and/or controlled by the same entity or by different entities. For example the owner of resource 110 could outsource the market-based allocation to a business partner operating resource agent 104, while keeping the network management and control function in its own hands, by retaining operation of the network control and management agent 108 themselves.

In the embodiment shown, network control and management agent 108 controls resource 110 to implement the allocation command received from resource agent 104. Thus, agent 108 commits the resource allocated to a player after the resource agent 104 has closed the bidding. Network control and management agent 108 provides a common interface for resource agent 104 for all systems supported. In certain embodiments, there is one agent 108 per resource. Alternately, a single agent can control multiple resources and communicate with multiple resource agents 104. In the embodiment shown, network management and control agent 108 controls resources belonging to other entities (i.e., not to the owner of resource agent 104). In other embodiments, resource 110 might also be under the direct control of the owner of resource agent 104.

Fig. 1 shows that network control and management agent 108 sends either Simple Network Management Protocol (SNMP) commands or COPS (Common Open Policy Service Protocol) commands to the resource. The SNMP protocol is well known and is described in RFC 1089, RFC 1270, RFC 1303, RFC 1298, RFC 1418, and RFC 1419, which are incorporated herein by reference in their entirety. The COPS protocol is well known and is described in "The COPS (Common Open Policy Service," dated March 5, 1999, available from Adobe Systems, Inc. of San Jose, CA, and RFC 2748., both of which are incorporated herein by reference in their entirety. Other appropriate resource control protocols can be used without departing from the spirit of the invention.

Once one or more winning buyers are determined, resource agent 104 alerts accounting agent 106, which keeps track of the winning buyers, as described below in connection with Fig. 9(a). Accounting agent 106 provides a common interface for all accounting systems supported. Agent 106 preferably contains a database handler for each system it supports. Although accounting agent 106is shown as using one or more of the IHN and SQL database interfaces, any appropriate interface to an accounting database could be used.

The 102 operates in accordance with one or more strategy rules for that agent. A strategy rule tells the what strategy to use in bidding against other agents for particular resources and, therefore constitutes a bidding mechanism for agents.

The s also operate in accordance with valuation rules that tell the how to value a particular resource when bidding (this value is often used as a part of the strategy rule). Thus, a valuation rule typically tells an agent how to value each unit of a resource over a range of possible quantities, at a given time. Seller agents also contain their own strategy and valuation rules, which allow them to decide how much of a resource to offer and how to determine a minimum price for the resource. Strategy and valuation can depend on external information, such as accounting information and network congestion.

The valuation and strategy of player agents (buyer and sellers) and resource agents are aware of a global allocation rule used by the resource agent to allocate a resource between the buyers. In the buyer and seller agents, this allocation rule is often considered in determining valuation and/or strategy. Thus, an allocation rule typically can be thought of as corresponding to a market mechanism. Examples of allocation rules include English auctions (familiar to persons who frequent human-based antique and estate sale auctions), Reverse Price Auctions (such as the main eBay model), Dutch auctions, and continuous bid-ask trading. Examples of allocation rules are found, for example, in 1) J.F. Rosenschein and G. Zlotkin, "Rules of Encounter," MIT Press 1994; 2) PhD thesis of N. Semret, "Market Mechanisms for Network Resource Sharing," Dept. of Electrical Engineering, Columbia University, submitted approximately May 1999; and 3) H.R. Varian, "Economic Mechanism Design for Computerized Agents," USENIX Workshop on Electronic Commerce, July 1995. Each of these three references is incorporated herein in its entirety.

Fig. 2 is a flow chart of a method performed by resource agent 104 of Fig. 1. In at least certain embodiments, there is more than one resource agent – one for each resource. For example, if an ISP offers several different bandwidths, each bandwidth might be considered a separate resource, controlled by a separate resource agent 104. Resource agents preferably run within servers that can be distributed over a network. Alternately, one or more resource managers 104 can reside on the same system.

As shown in Fig. 2, resource agent 104 receives 202 one or more bids for resource 110 from one or more s 102. Such bids will usually include at least quantity and price values. If a buyer is outbid, the resource agent notifies 204 the buyer, unless the trading period is over 206. Once the trading period is over, the resource agent notifies 208 the

winning or agents and proceeds to send an allocation command 210 to network control and management agent 108 that will cause agent 108 to control the resource in accordance with the winning bids.

Fig. 3 is a flow chart of a method performed by player/buyer 102 agent of Fig. 1. First, the is apprised of potential resources available for bidding. This is accomplished either by the player requesting current resource auction information from a centralized directory service (not shown) or by the player registering with the resource agent and periodically being sent information about current bidding. With a directory service, a player agent 102 queries the directory service to find the location (e.g., in the form of a URL) of resource agents 104, garages (see Fig. 10), and other player agents 102. The decides 302 whether to bid on a particular unit or resource 110 and sends 304 a bid to the resource agent. If the receives a notice that he has been outbid, control returns to element 302 and the agent decides whether to continue bidding.

In element 302, the uses its knowledge of the system market allocation rule and of its own valuation rules to determine how much it is willing to pay for resources at a given time and uses its own strategy rules to determine whether to bid on available resources. If the includes a GUI, a human being can visualize the market using various known graphs, charts or similar graphics. A user can also use the GUI to change the strategy and valuation rules used by the agent.

It should be noted that certain embodiments include a special type of player/, called a broker agent, which buys resources with the intent of reselling those resources.. Thus, for example, a broker agent has no user for the resource (such as bandwidth) itself, since the broker is not an ISP or similar entity in need of bandwidth. Instead, the broker arranges for third party customers, such as ISPs, to receive the benefit of the resources the broker has bid for and won, by reselling them through another resource agent instance, and the broker keeps the difference between the buying and selling price.. In such a case, the new resource agent 104 informs the network control and management agent 108 which party is to benefit from the resource bid for by the broker. Fig. 4 is a flow chart of a method performed by a player/seller 102 agent of Fig. 1. A seller agent first notifies 402 the resource agent that it has one or more units of a resource to sell (for example, 1Mbs for 5 minutes or 10 minutes of processor time). Once the bidding is over, seller agent 102 will receive from resource agent 104 a notification 404 of the winning bidder or bidders. In at

least one embodiment, the seller agents collect the payments from the buyers based on accounting information retrieved from the accounting agent. 106.

Fig. 5 is a chart showing an example of the result of a Progressive Second Price Auction (PSP) allocation rule. Other allocation rule types, as discussed above, can also be used, as specified by the particular allocation market strategy implemented in a particular system. Fig. 5 shows how resources are allocated and prices set once the bidding is over and it is determined that there are not enough resources to satisfy all the bidders.

In the PSP auction of Fig. 5, bidder A bids $3 for 50 resource units; bidder B bids $2 for 30 resource units; bidder C bids $1 for 30 resource units; and bidder D bids $0.50 for 20 resource units. At the close of bidding, resource agent 104 allocates the resources as follows: The 100 available resource units are apportioned between the bidders until the resource is gone. Thus, the high bidder A is allocated all of the resource that he wants, as is the second high bidder B. Bidder C only gets 20 or the 30 resource units that he wanted and bidder D gets nothing, since there are no more resource units to allocate after partially fulfilling bidder C's wants.

The resource agent 104 determines the amount that the bidders in the PSP auction are charged as follows: For each bidder, the resource agent determines the value of the bidder's resource if that bidder had not participated, and charges the bidder a price based on this determination.

Thus, if bidder A had not participated, his 50 units would have been allocated as follows:

10 units to bidder C (to make up C's shortfall)

20 units to bidder D (to give D his requested number of units)

The remaining 20 of bidder A's units would not have been allocated.

The cost to bidder A is determined as follows:

The cost of a resource unit to bidder C is $1/30.

Thus, the value that would have been given to bidder C: 10 x ($1/30) = $0.33

The cost of a resource unit to bidder D is $0.50/20.

Thus, the value that would have been given to bidder D: 20 x ($0.50/20) = $0.50

Thus, the cost of A's resources had A not participated is $0.33 + $0.50 = $0.83. Bidder A is charged this amount for his 50 units of resource.

Similarly, if bidder B were not present, his 30 units would have been allocated as follows:

10 units to bidder C (to make up C's shortfall)

20 units to bidder D (to give D his requested number of units)

The cost to bidder B is determined as follows:

The cost of a resource unit to bidder C is $1/30.

Thus, the value that would have been given to bidder C: 10 x ($1/30) = $0.33

The cost of a resource unit to bidder D is $0.50/20.

Thus, the value that would have been given to bidder D: 20 x ($0.50/20) = $0.50

Thus, the cost of B's resources had B not participated is $0.33 + $0.50 = $0.83. Bidder B is charged this amount for his 30 units of resource.

Similarly, if bidder C were not present, his 20 units would have been allocated as follows:

20 units to bidder D (to give D his requested number of units)

The cost to bidder C is determined as follows:

Value that would have been given to bidder D: 20 x ($0.50/20) = $0.50

Thus, the cost of C's resources had C not participated is $0.50. Bidder C is charged this amount for his 20 units of resource.

It should be noted that Fig. 5 shows only how resources are allocated after bidding is closed. An allocation rule also includes within it rules or explanations of how the auction itself should be conducted. For example, a PSP auction generally lasts for a predetermined amount of time (for example, five minutes). While the bidding is open, resource agent 104 collects all bids received from the s 102 and saves them in a bidlist data structure (e.g., a

linked list). The bidlist data structure indicates which bid is the most recent bid for each 102.

As bids are received from the s 102 by the resource agent 104, the resource agent 104 transmits the bids received to the other agents 102, so that all agents know what all other participating agents are bidding. Because each 102 has knowledge of the allocation method, each 102 can apply its strategy and valuation rules to determine whether that agent is going to be allocated the resources on which it has bid. The agent, applying the allocation, strategy, and valuation rules, determines whether it should bid again. In a PSP auction, bidding usually stabilizes after a few minutes. In some embodiments, resource agent 104 does not hold the auction open for a predetermined time, but instead waits a predetermined amount of time after the bidding has stabilized to make sure that no other bids are received. In some cases, resource agent 104 announces to the s 102 that bidding will close in a certain number of minutes or seconds. In some cases, if a bid is received during this time period, the auction is kept open a bit longer.

Thus, an allocation rule (known to all agents) also includes information about how the auction will be conducted by resource agent 104, including, for example: how long an auction will last, if the auction has no set time period, what are the conditions for the auction to close. In addition, as described above, the allocation rule includes rules describing how price and quantities are assigned after the auction has closed.

In a preferred embodiment of the invention, auctions last 5 minutes, although other periods of time could be used and these periods of time could be either variable or user-settable. In a preferred embodiment, the bandwidth resource being auctioned during a current auction is allocated immediately and another auction is begun immediately. Thus, an auction occurs roughly every five minutes for the bandwidth that will be used by the buyers during the next five minutes. Other embodiments may not auction all bandwidth for immediate use.

The PSP auction model is described further in A.A. Lazar and N. Semret, "Design and Analysis of the Progressive Second Price Auction for Network Bandwidth Sharing," Telecommunications Systems, Special issue on Network Economics, which is attached hereto as Appendix A and forms a part of this application.

Other examples of allocation rules include, but are not limited to a Hold Option Auctions, which are discussed, for example, in the PhD thesis of N. Semret, "Market Mechanisms for Network Resource Sharing," Dept. of Electrical Engineering, Columbia

University, submitted approximately May 1999, Chapter 4 of which (28 pages) is attached hereto as Appendix B and which forms a part of this specification. The Hold Option is a concept for advance price and quantity guaranteed reservations of network resources in a real-time market environment. Periodic auctions (progressive second price, or other) among arrivals grouped in batches give rise to the spot market of capacity changes. A reservation guaranteeing access for an arbitrary duration with a capacity piece below the bid can be made at any time before or during service. This eliminates the risk (which is inherent on the spot market) of losing resources to higher bidders before service completion. The reservation is defined as a hold option, and is analogous to derivative financial instruments such as options and futures integrated over time. Based on a heavy traffic diffusion model, reservation fees can be computed as the fair market price of a hold option. In at least one embodiment, special player agents 102 are allowed to place such hold options, thus providing a guaranteed reservation of a network resource for an arbitrary duration.

Figs. 6(a) and 6(b) show examples of valuation rules. In Fig. 6(a), a valuation rule is formed of pairs of quantity /price values. In Fig. 6(b), a valuation rule is formed as a function of various input variables. These input variables can include any or (but are not limited to): the number of hits on a web site ; the average file size downloaded (and the bandwidth needed to service those files); the expected delay or expected latency, and the value of each hit. Valuation can also be time dependent (e.g., higher valuations are assigned during peak usage times when more bandwidth is needed) and the network state (e.g., more bandwidth is needed if the network is congested).

It should be noted that there are engineering tradeoffs for the type of valuation rule used. Because a low-level valuation rule requires more inputs, which require more time and effort to collect and receive, a low-level description may require a large amount of data to be transferred in order for the agent to be able to bid in accordance with the low-level valuation rule. On the other hand, a simple, high-level valuation rule reduces the ability of an agent to make an optimum bid because the agent is operating with less information. Either implementation can be correct for a given circumstance, depending on the needs of the particular player agent and the limitations of its system and network.

Figs. 7(a) and 7(b) shows examples of strategy rules. Fig 7(a) shows a simple rule set, where the first precedent is to identify the type of allocation system being used. Once the allocation system if identified, the 102 applies a set of predefined conventional rules to determine whether it should bid (or bid again). Fig. 7(b) shows an example where the

strategy is based on a user-defined function. In the example, if the function reaches a threshold value, the agent 102 will bid (or bid again). Other examples of strategy rules decide not just whether the agent should bid, but how much the agent should bid, in accordance with the allocation rule being used by the system and in accordance with factors specific to that agent (such as, for example, those factors discussed above in relation to valuation rules).

Certain strategy rules are very simple and involve bidding constant amount. Such simple strategy rules may result in uneven amounts of bandwidth being won. Another example strategy rule is periodic bidding, in which a buyer agent enters auctions periodically.

Fig. 8 shows an example of a flow chart used by an accounting system of an embodiment of the invention. The accounting system is notified whenever a resource agent closing bidding on a resource unit and keeps track 802 of the winning bids and resulting allocations. Periodically, the accounting system bills 804 the seller a percentage of the resources sold, which is received by the owner of the resource agent for operating the resource agent and account agents of the Merkato platform.

Fig. 9(a) is a more detailed block diagram of the embodiment of Fig. 1. It will be understood that Fig. 9(a) details only one possible way to implement the present invention and that the description herein is not to be taken in a limiting way with regard to operating systems, protocols, programming languages, etc. The example shown is implemented in Java using the World Wide Web, but the invention is not limited to such a system. In fact, the invention can be implemented on any appropriate computers and networks, using any appropriate programming language, hardware, software, and operating system. Parts of the system can be implemented in software, firmware, or hardware, as needed.

Fig. 9(a) includes four layers: an operating system (OS) layer 902, a Java layer 904, a Merkato layer 906, and a diffex layer 108. In the described embodiment, the OS layer 902 and the Java layer 904 are conventional and will not be described herein. Other embodiments may make changes to one or more of layers 902 and 904 to enhance the performance of the system, for example. In this example, the use of Java layer 904 makes the system platform independent. Thus, the Merkato layer 906 can run on any computer or computing device (such as a wireless device , pager, cell phone, or Internet appliance)

Layer 906 allows a player agent 102 to be executed on the client side, from a Java application or an applet executing in a Web browser. Alternately (or in addition), a player

agent 102 can be executed as a servlet on a web server, which is the "garage" environment of Fig. 10.

Layer 908 enables real-time resource markets between peering ISPs. The layer 908 allows ISPs to buy and sell resources, such as bandwidth, from each other in real-time. Layer 908 auctions in real-time the outgoing bandwidth on each ISP's line out of the exchange, ensuing that at all times, the bandwidth goes to the buyer with the highest value for it. Layer 908 also allows for buying and selling in advance (i.e., making reservations with guaranteed capacity and capped prices) through a derivative market of options, in effect enabling the trading of risk and hedging, for original ISP buyers and sellers, as well as for purely financial players.

Fig. 9(b) shows and example of how a winning buyer ISP 952 is given a resource, such as bandwidth. In Fig 9(b), a seller ISP 956 has a seller 902 agent running on a diffex client. As discussed above, the seller agent determines that the ISP has bandwidth to sell (e.g., through user input via the seller agent GUI) and makes that bandwidth available for auction by sending a message to resource agent 904 (see numeral 1 in a circle). The buyer ISPs 952, 954 have s 902 running on a diffex client. The s bid on the bandwidth in accordance with their allocation rules, strategy rules, and valuation rules, as discussed above (see numeral 2 in a circle). For certain types of allocation models, the agents also receive information about bids during the auction (not shown). Once the auction is concluded, resource agent 904 sends an allocation command to network control and management agent 908 (numeral 3 in a circle), which in turn controls a router 970 of the seller ISP so that the winning buyer ISP receives its bandwidth (see numeral 4 in a circle).

Resource agent 104 in layer 908 interfaces with the resource (e.g., with the router 970 of the seller ISP) to allocate the bandwidth to the winning bidder. Specifically, resource agent 904 interfaces with control agent 908 to send commands to the router of the seller ISPs. If, for example, the resource is bandwidth, allocation to the winning bidder can be effected by one of the following or by any other appropriate method:

a) the router 970 is given a set of class-based weighted-fair queuing parameters to be used by the router to control packet queuing in the router so that the winning ISP buyers are assured of receiving the bandwidth which they was allocated by the resource agent. These queuing parameters will give priority to the winning bidders when packets from the winning ISPs are sent to the seller ISP's router.

b) the router is given a set of committed access-rate parameters to be used by the router to limit and shape traffic assure each buyer the bandwidth which it is allocated, or

c) capacity within an MPLS (Multiprotocol Label Switching) tunnel between two points in the seller's network.

Thus, as shown in Fig. 9(b), packets 960 from the winning buyer ISP(s) are routed through the seller's router and on to the seller ISP's network 956, from which they are delivered to their destination. In at least one embodiment, the winning ISPs are aware that they have won and use existing routing protocols to ensure that they direct packet traffic to the seller ISP's router. In other embodiments, the resource agent informs routers in the winning ISPs of the needed routing change using known routing protocols.

In the described embodiment, the resource agent is always executed on a Web server. Each resource agent 104 runs as a servlet on a Web server. This architecture is scalable because resource agents 104 can be distributed to run on any Web servers supporting the concept of a servlet.

In the described embodiment, communications between a player agent (either a buyer or a seller) 102 and a resource agent 104 are performed via a resource agent proxy using any of http extensions, native TCP protocol, or Java's remote method invocation (rmi). All communications are secured through an appropriate mechanism such as the secured socket layer (SSL). A shown in Fig. 9(a), each agent (player and resource) has an allocation rule object 128. The player agents 102 also have valuation rule objects 124, strategy rule objects, 126, and a GUI object 122. In addition, the resource agents 104 have networking and accounting drivers for interfacing with external support systems. Further security is provided through the implementation of specific allocation rules that protect the stability of the system. Specifically, each user is required to pay a bid fee to resource agent 104 for every bid sent. The implementation of a bid fee is intended to prevent users from trying to artificially destabilize the resource price and to prevent a "man in the middle" attack, which is defined as a third party intercepting a bid from another agent and keeping sending the same bid over and over. In addition, timestamps are preferably used to discriminate every bid. Thus, if a bid has a previously used timestamp, resource agent 104 will ignore that bid.

In the described embodiment, communication between agents is effected using http, thus bypassing many problems associated with firewalls, proxies, and other middleware.

Fig. 10 shows an example of an embodiment of the present invention including a "garage" for player agents 102'. The garage 130 is a component on a web server that serves the purpose of storing agents and enabling their execution remotely from a user's computer. The garage contains an "attendant" program 131 whose job is to help mobile agents find a parking place in the garage and to ensure proper agent execution when the agents want to run autonomously. Garage 130 performs the function of a distributed database to store the agents 102' and provides a distributed processor (not shown) to execute the agents. Garage 130 can be located on the same system as resource agent 104, but can also be located on a different machine.

The attendant is an example of a multi-agent. Multi agents coordinate multiple simple agents to act together. Multi-agents can be used to form coalitions of agents, using the attendant to bid for aggregated resources on behalf of the coalition. In the described embodiment, the agents communicate with the attendant to let the attendant know that they need resources. The multi-agent aggregates the various types of resources requested (e.g., different connection speeds or different bandwidths) and uses the allocation rule and its own strategy and valuation rules to bid on behalf of the agents. If the multi-agent wins, the resource is divided between the agents and the attendant so informs the resource agent 104, which allocates the bandwidth accordingly. A broker is an example of a multiagent, coordinating buyer and seller agents for a common objective to act as a profit-maximizing reseller.

In the described embodiment, agent mobility to the garage is provided through XML. The syntax and semantics of an agent is described using XML. For example, the semantics of an agent can include its allocation, strategy and valuation rules. Each agent can be transferred to the "garage" 130 by generating its XML description. Once an agent is provided in this form, it can be re-instantiated either in a garage 130 or in a user's computer in an applet or a java application.

Figs. 11(a)-11(b) are an example of an XML file for a generic player agent 102. This XML would be used, for example, to send the agent 102 to a client for execution in garage 930. Each of the XML tags (indicated by <> brackets) identifies an attribute of the agent that is to be activated in the garage. It will be understood that the XML of Fig. 11 is shown for purposes of example only and that other embodiments of the invention may use more or fewer tags and/or different tags than those shown in Fig. 11.

Figs. 12(a)-12(c) shows an example of Java code implementing a strategy rule for a 102. This example implements a strategy in accordance with the PSP auction model described above. As shown in section 1234 of Fig. 12(c), an agent using this strategy will submit a new bid only if the new bid determined in the rule is increased by at least a calculated value epsilon. Note that this strategy rule calls a valuation rule in line 1232 of Fig. 12(c). This valuation rule implements a PSP valuation method.

Figs. 13(a)-13(c) show an example of Java code implementing an allocation rule for a resource agent 104. This example looks at a number of received bids in a bidlist data structure and computes an allocation (similar to that of Fig. 5) given the current bids on the bid list in accordance with a PSP auction allocation model.

Fig. 14 is an example of an XML file for a generic resource agent 104. This XML would be used, for example, to send the resource agent 104 to a web server. Each of the XML tags (indicated by <> brackets) identifies an attribute of the agent that is to be activated on the server side.

Figs. 15(a)-15(r) show examples of user interfaces for buyer and seller agents. Figs. 15(a) and 15(b) show an example of an html GUIs that provides static information to a user. Figs. 15(c) and 15(d) show an example of GUIs implemented as a Java applet. The information provided by these interfaces is continuously updated in real-time or periodically. Figs. 15(e)-15(r) show an example of an advanced GUI, which is preferably also implemented as an agent. Particular buyer and seller agents may have one, none, or all of the particular GUIs shown here, or may have GUIs providing other relevant information not shown here. Note that several of these GUIs allow a human user to choose the valuation and/or strategy rules and to determine when and how to bid. It should be understood that in the preferred embodiment, buyer agents are capable of bidding on their own. Other embodiments may contain agents that bid only at the direction human beings.

Accordingly, the present invention is intended to embrace all such alternatives, modifications and variations as fall within the spirit and scope of the appended claims and equivalents.

APPENDIX B

Hold Option

Docket Number: 61624-04980

What is claimed is:

1. A method for real-time market-based resource allocation, comprising:
receiving at least one bid in real-time for a resource;
deciding which of the at least one bids has won the bidding; and
controlling the resource so that it is committed to the winning bidder.

2. The method of claim 1, wherein the resource is bandwidth.

3. The method of claim 1, wherein the resource is buffer space.

4. The method of claim 1, wherein the resource is processor time.

5. The method of claim 1, wherein the resource is controlled in real-time so that the winning bidder has access to the resource he has won as soon as bidding closes.

6. The method of claim 1, wherein the bid is received from a software agent.

7. The method of claim 6, wherein the software agent bids in accordance with a strategy rule.

8. The method of claim 7, wherein the strategy rule is a truthful best reply strategy.

9. The method of claim 6, wherein the software agent bids in accordance with a valuation rule.

10. The method of claim 9, wherein the valuation rule is determined in accordance with at least one measured network parameter.

11. The method of claim 6, wherein the software agent bids in accordance with an allocation rule.

12. The method of claim 1, wherein deciding which bid has won the bidding is performed in accordance with an allocation rule.

13. The method of claim 12, wherein deciding which bid has won the bidding is performed in accordance with a market allocation rule also used by a buyer software agent, the market allocation rule defining the rules of the resource market.

14. The method of claim 12, wherein deciding which bid has won the bidding is performed in accordance with an English Auction market allocation rule.

15. The method of claim 12, wherein deciding which bid has won the bidding is performed in accordance with a continuous bid-ask trading market allocation rule.

16. The method of claim 12, where deciding which bid has won the bidding is performed in accordance with a progressive second price auction allocation rule.

17. The method of claim 12, where deciding which bid has won the bidding is performed in accordance with a hold option allocation rule.

18. The method of claim 1, further comprising:

storing information concerning which bid has won, for accounting purposes.

19. The method of claim 1, wherein there are several resources, and a respective resource agent performs the elements of claim A for each resource.

20. The method of claim 1, wherein there is one resource, and a single resource agent performs the elements of claim 1 for the resource.

21. The method of claim 1, wherein the elements of claim 1 are performed by a resource agent executing on a separate computer than the placing the bids.

22. The method of claim 1, wherein the elements of claim 1 are performed by a resource agent executing on the same computer as at least one placing at least one of the bids.

23. The method of claim 1, wherein the is controlled by a human being, who decides how to bid.

24. The method of claim 1, further comprising:

deciding, by a bidder agent, how to bid based on a valuation function of the bidder agent and a strategy algorithm of the .

25. The method of claim 1, further comprising: receiving an offer to sell resources from a seller agent.

26. The method of claim 1, further comprising:

receiving player agents in a garage of a resource agent and receiving bids from player agents in the garage.

27. The method of claim 1, wherein the bid is received from a player agent that also submits offers to sell resources.

28. The method of claim 1, further comprising:

before bidding, receiving a request from a buyer software agent for a location of resource agent.

29. The method of claim 1, further comprising:

sending at least one bid in real-time for a resource;

receiving a notification that the sent bid has won the bidding;

selling the use of the winning resource to third parties through a separate resource agent; and

notifying a resource management and control agent that the third parties will be using the resource.

30. A method, performed by a multiagent, comprising:

sending at least one bid in real-time for a resource in accordance with a strategy rule and a valuation rule of the multiagent and in accordance with a system-wide allocation rule;

receiving a notification that the sent bid has won the bidding;

selling the use of the winning resource to third parties through a separate resource agent; and

notifying a resource management and control agent that the third parties will be using the resource.

31. A method, performed by a buyer agent, comprising:

sending at least one bid in real-time for a resource in accordance with a strategy rule and a valuation rule of the buyer agent and in accordance with a system-wide allocation rule;

receiving a notification that the sent bid has won the bidding; and

making use of the resource in real time, immediately after winning the bid.

32. The method of claim 32, wherein the system wide-allocation rule is a PSP allocation rule.

33. A system for real-time market-based resource allocation, comprising:

means for receiving at least one bid in real-time for a resource;

means for deciding which of the at least one bids has won the bidding; and

means for controlling the resource so that it is committed to the winning bidder.

34. A system for real-time market-based resource bidding by a multiagent, comprising:

means for sending at least one bid in real-time for a resource in accordance with a strategy rule and a valuation rule of the multiagent and in accordance with a system-wide allocation rule;

means for receiving a notification that the sent bid has won the bidding;

means for selling the use of the winning resource to third parties through a separate resource agent; and

means for notifying a resource management and control agent that the third parties will be using the resource.

35. A system for real-time market-based resource bidding by a buyer agent, comprising:

means for sending at least one bid in real-time for a resource in accordance with a strategy rule and a valuation rule of the buyer agent and in accordance with a system-wide allocation rule;

means for receiving a notification that the sent bid has won the bidding; and

means for making use of the resource in real time, immediately after winning the bid.

36. The system of claim 35, wherein the system wide-allocation rule is a PSP allocation rule.

37. A system for real-time market-based resource allocation, comprising:

a portion configured to receive at least one bid in real-time for a resource;

a portion configured to decide which of the at least one bids has won the bidding; and

a portion configured to control the resource so that it is committed to the winning bidder.

38. A system for real-time market-based resource bidding by a multiagent, comprising:

a portion configured to send at least one bid in real-time for a resource in accordance with a strategy rule and a valuation rule of the multiagent and in accordance with a system-wide allocation rule;

a portion configured to receive a notification that the sent bid has won the bidding;

a portion configured to sell the use of the winning resource to third parties through a separate resource agent; and

a portion configured to notify a resource management and control agent that the third parties will be using the resource.

39. A system for real-time market-based resource bidding by a buyer agent, comprising:

a portion configured to send at least one bid in real-time for a resource in accordance with a strategy rule and a valuation rule of the buyer agent and in accordance with a system-wide allocation rule;

a portion configured to receive a notification that the sent bid has won the bidding; and

a portion configured to make use of the resource in real time, immediately after winning the bid.

40. The system of claim 39, wherein the system wide-allocation rule is a PSP allocation rule.

41. A computer program product, including instructions for causing a data processing device to perform actions for real-time market-based resource allocation, comprising:

receiving at least one bid in real-time for a resource;

deciding which of the at least one bids has won the bidding; and

controlling the resource so that it is committed to the winning bidder.

42. A computer program product, including instructions for causing a multiagent of a data processing device to perform actions for real-time market-based resource bidding, comprising:

sending at least one bid in real-time for a resource in accordance with a strategy rule and a valuation rule of the multiagent and in accordance with a system-wide allocation rule;

receiving a notification that the sent bid has won the bidding;

selling the use of the winning resource to third parties through a separate resource agent; and

notifying a resource management and control agent that the third parties will be using the resource.

43. A computer program product, including instructions for causing a buyer agent of a data processing device to perform action for real-time market-based resource bidding, comprising:

sending at least one bid in real-time for a resource in accordance with a strategy rule and a valuation rule of the buyer agent and in accordance with a system-wide allocation rule;

receiving a notification that the sent bid has won the bidding; and

making use of the resource in real time, immediately after winning the bid.

44. The computer program product of claim 43, wherein the system wide-allocation rule is a PSP allocation rule.

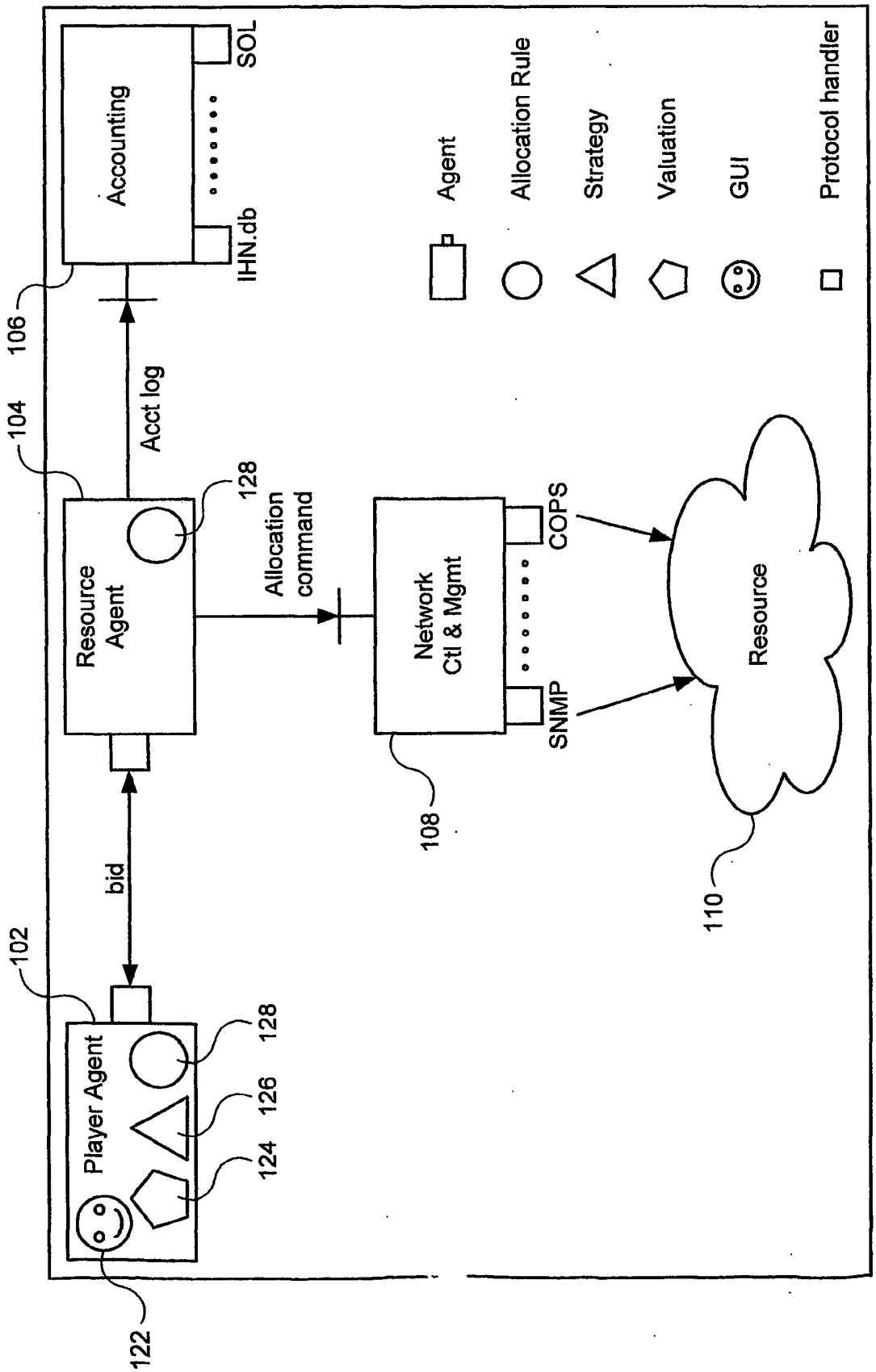
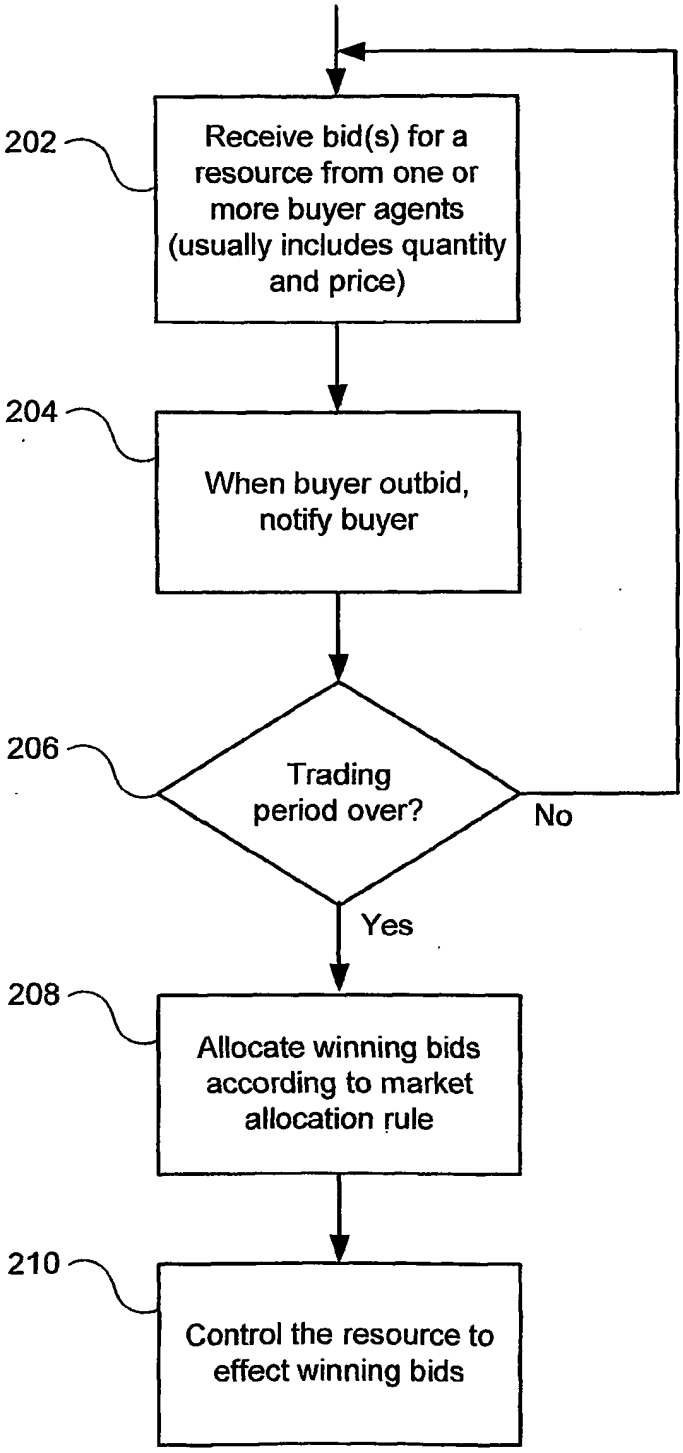
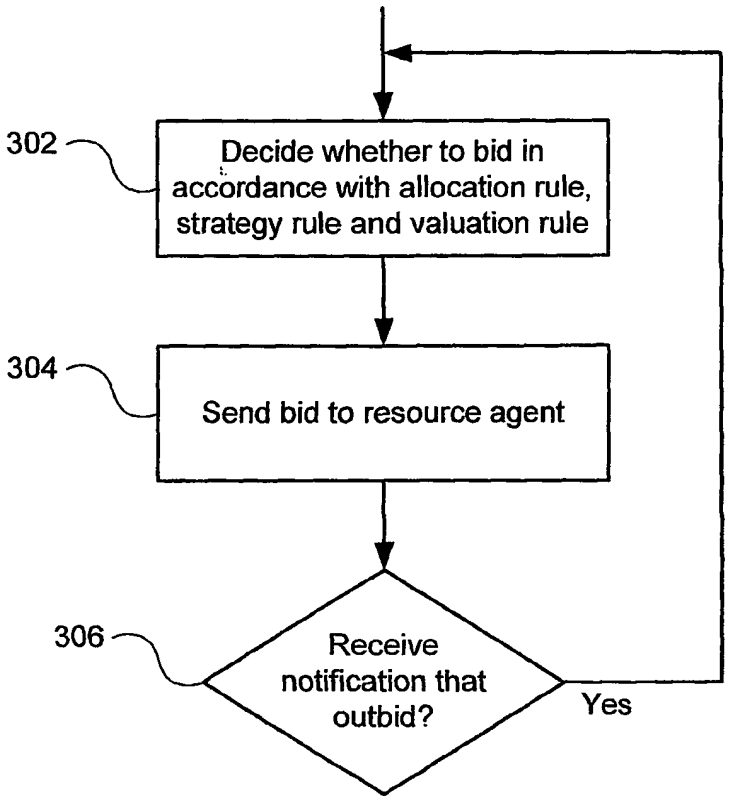

Accounting
SQL
IHN.db
106
104
Acct log
128
Resource Agent
Allocation command
Network Ctl & Mgmt
COPS
SNMP
108
Resource
110
bid
102
Player Agent
128
126
124
122
Agent
Allocation Rule
Strategy
Valuation
GUI
Protocol handler

Figure 1



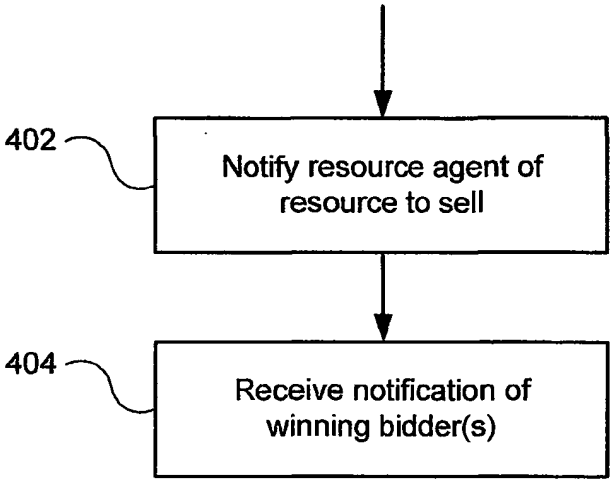
Resource Agent

Figure 2



Player Agent (Buyer)

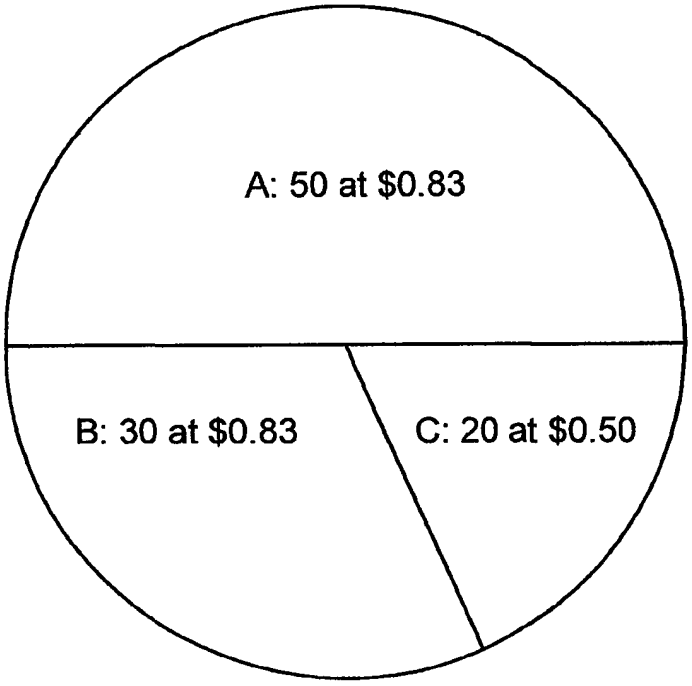
Figure 3



Player Agent (Seller)

Figure 4

100 units of resource:
A bids for 50 at $3,
B bids for 30 at $2
C bids for 30 at $1
D bids for 20 at $0.50



Example Market Allocation Rule (PSP)

Figure 5

Quantity	Price	Valuation
—	—	—
—	—	—
—	—	—
—	—	—

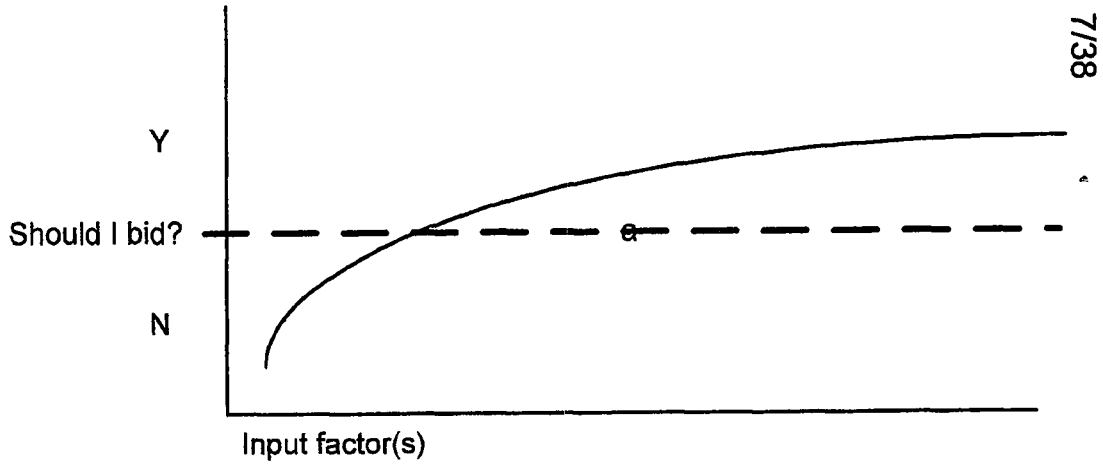
Example Valuation Rule
Figure 6(a)

Example Valuation Rule
Figure 6(b)

If allocation rule is PSP
[actions for PSP bidding]

If allocation rule is English auction
[actions for English auction]

Example Strategy Rule
Figure 7(a)



Example Strategy Rule
Figure 7(b)

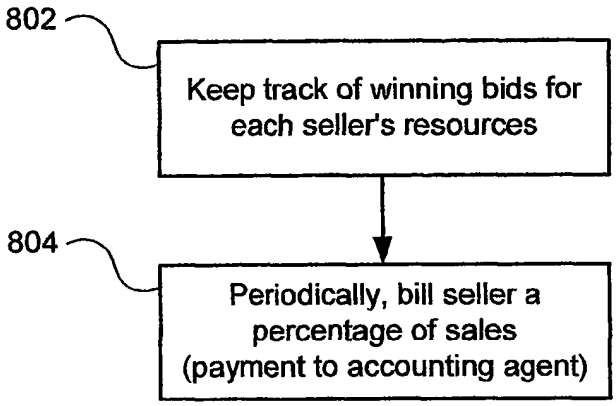

802
Keep track of winning bids for each seller's resources
804
Periodically, bill seller a percentage of sales (payment to accounting agent)

Figure 8

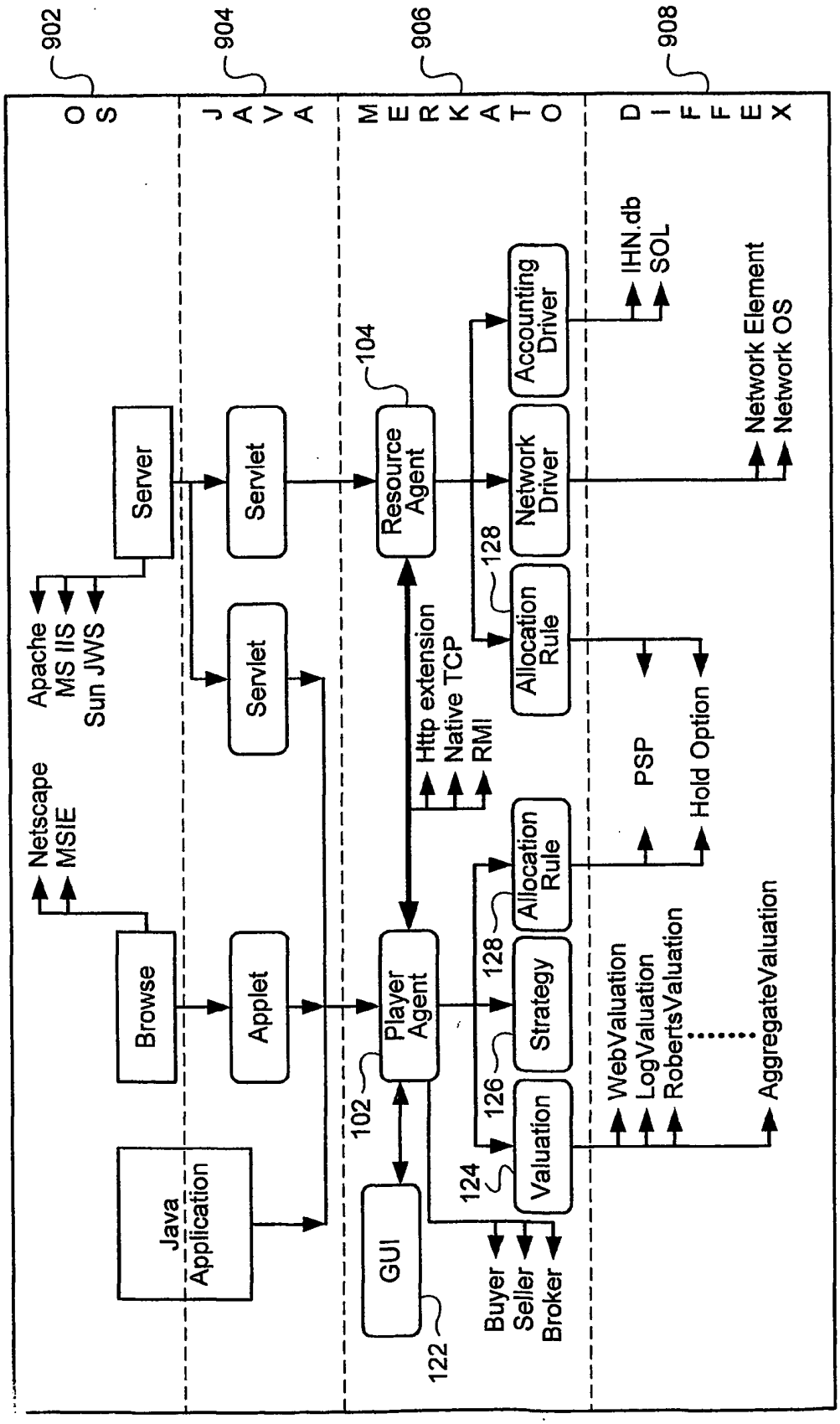

902
904
906
908
O S
J A V A
M E R K A T O
D I F F E X
Server
Servlet
Servlet
Servlet
Resource Agent
104
Accounting Driver
Network Driver
Allocation Rule
128
Allocation Rule
128
Strategy
126
Valuation
124
Player Agent
102
Browse
Applet
Java Application
GUI
122
Apache
MS IIS
Sun JWS
Netscape
MSIE
Http extension
Native TCP
RMI
IHN.db
SOL
Network Element
Network OS
PSP
Hold Option
WebValuation
LogValuation
RobertsValuation
AggregateValuation
Buyer
Seller
Broker

Figure 9(a)

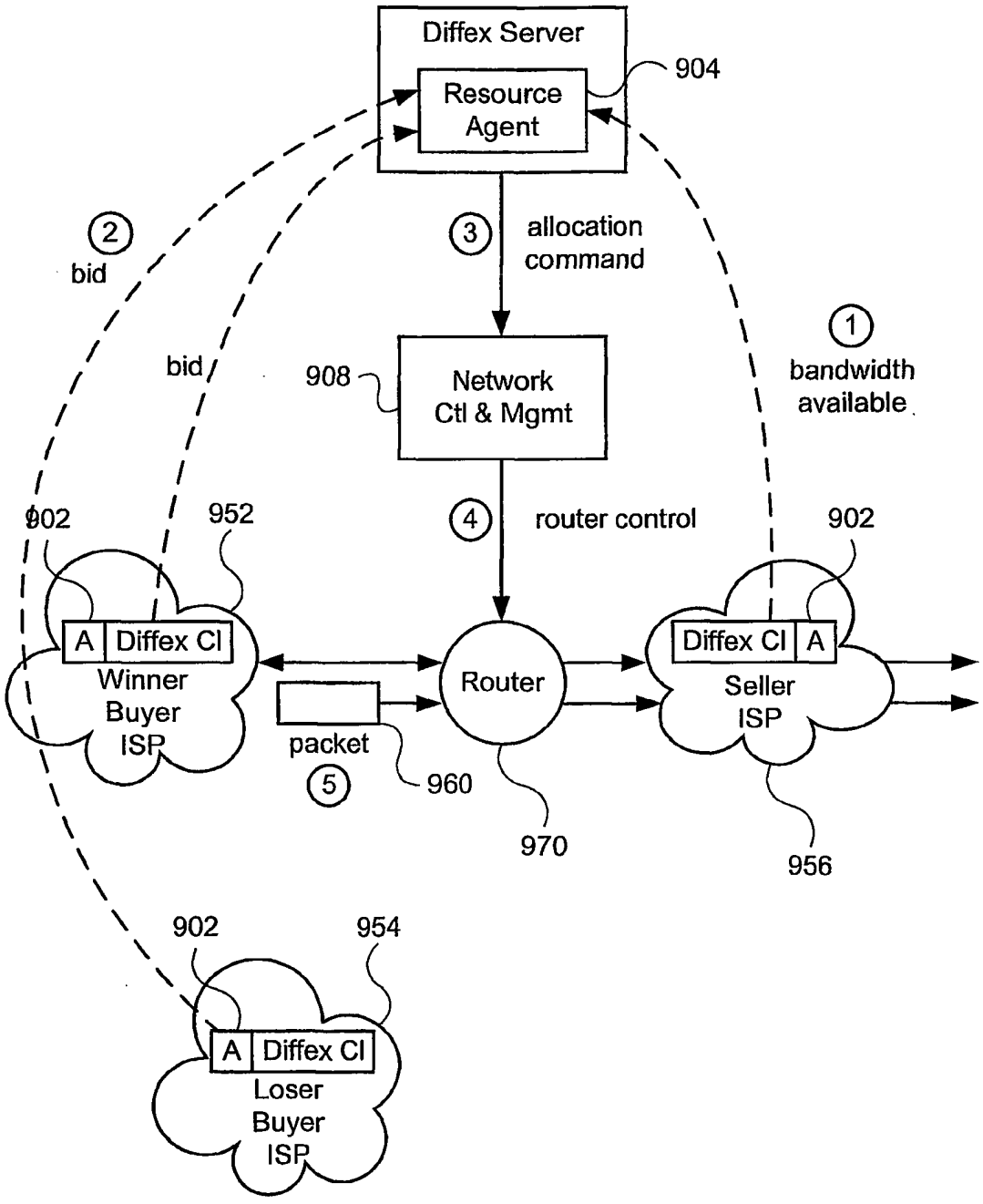

Diffex Server
Resource Agent
904
2
bid
3
allocation command
1
bandwidth available
bid
908
Network Ctl & Mgmt
4
router control
902
952
902
A
Diffex Cl
Winner Buyer ISP
Router
Diffex Cl
A
Seller ISP
packet
5
960
970
956
902
954
A
Diffex Cl
Loser Buyer ISP

Figure 9(b)

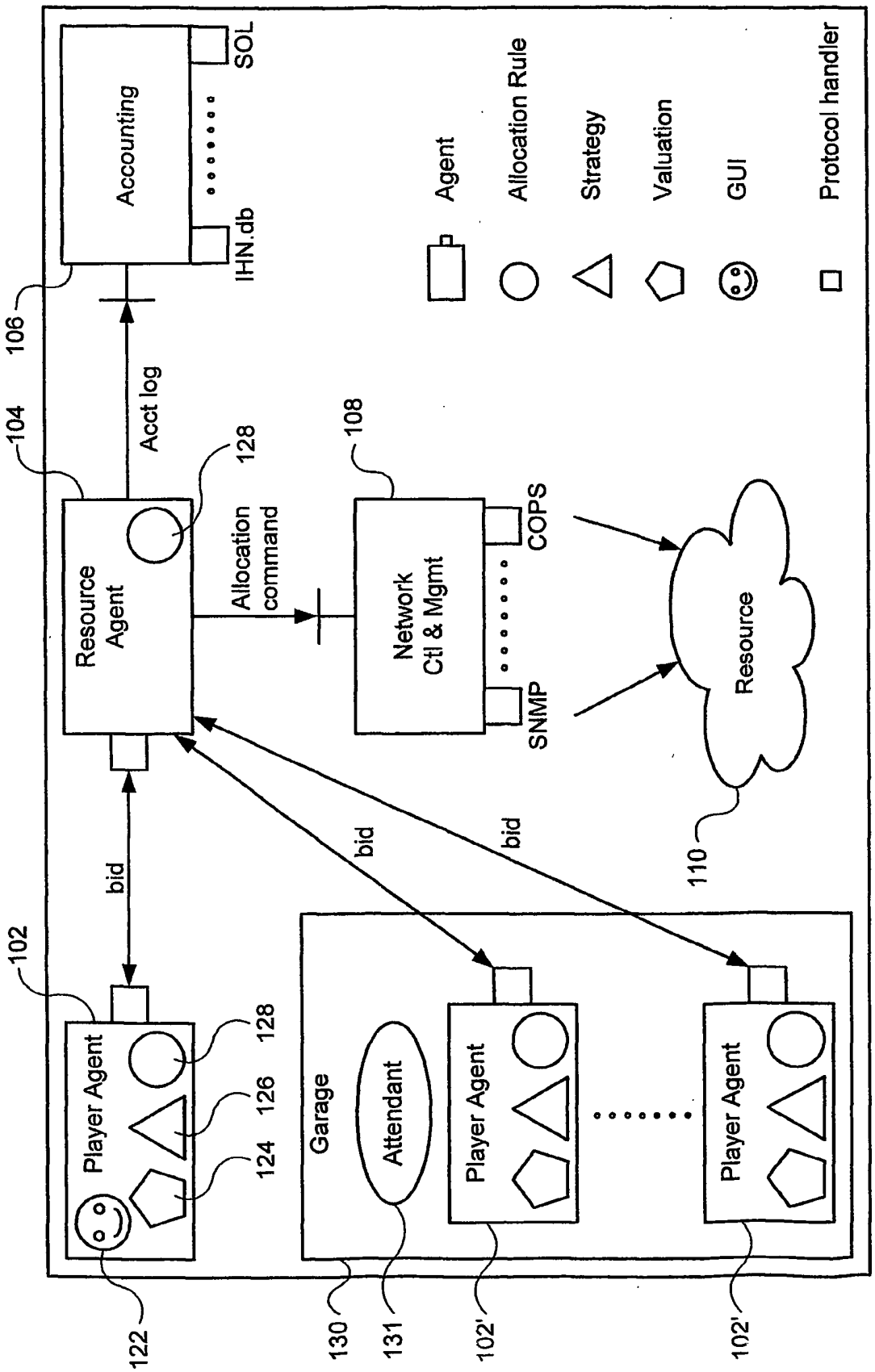

Accounting
SQL
IHN.db
106
104
Acct log
128
108
Resource Agent
Allocation command
Network Ctl & Mgmt
COPS
SNMP
Resource
110
bid
bid
bid
102
Player Agent
128
126
124
Garage
Attendant
Player Agent
Player Agent
122
130
131
102'
102'
Agent
Allocation Rule
Strategy
Valuation
GUI
Protocol handler

Figure 10

```
<?xml version="1.0" encoding="UTF-8" ?>
- <AuctionPlayer context="http://HOSTNAME:HTTP_PORT/bx/garage">
 - <SingleFrameGUI>
      <TextPanel name="News" height= "50" visible="true" border="false" />
      <LoginPanel name="Login" height="160" visible="true" border="true" />
      <ResourceAgentPanel name="ResourceAgent" height="80" visible="true"
       border="true" />
      <UploadAgentPanel name="Garage" height="80" visible="true"
       border="true" />
      <BidCanvasPanel name="BidCanvas" height="180" visible="false"
       border="true" />
    - <StrategyChoicePanel name="Strategies" height="160" visible="true"
       border="true">
       <StrategyPanel name="Manual" strategy="ManualStrategy" />
       <StartegyPanelNotEditable name="Auto" strategy="TruthfulStrategy" />
      </StrategyChoicePanel>
    - <ValuationChoicePanel name="Valuations" height="240" visible="false"
       border="true">
       <WebValuationPanel name="Web Valuation" valuation="WebValuation" />
       <ValuationPanel name="Elastic Demand" valuation= "RobertsValuation" />
       <ValuationPanel name="Inelastic Demand" valuation="LinearValuation" />
       <BudgetValuationPanel name="Budget Valuation" label=" "
        valuation="BudgetValuation" />
      </ValuationChoicePanel>
      <PlayerInfoPanel name="Allocation" height="120" visible="true"
       border="true" />
      <BudgetPanel name="Budget" height="80" visible="false" border="true" />
      <DisplayPanel name="Units" height="80" visible="false" border="true" />
      <IPAddressPanel name="IP" height="110" visible="false" border="true" />
      <ConnectionPanel name="Connection" height="140" visible="false"
       border="true" />
      <BidTablePanel name="Bid Table" height="400" visible="false"
       border="true" />
      <BidGraphPanel name="Bid Graph" height="400" visible="false"
       border="true" />
      <AllocationGraphPanel name="Allocation Graph" height="400" visible="false"
       border="true" />
   </SingleFrameGUI>
   <PlayerIdentity name="USERNAME" passwd="PASSWD" ipaddress="IP_ADDRESS"
    netmask="NETMASK" />
  - <LinearValuation label="Inelastic Demand">
     <Parameter name="qmax" value="45000.0" label="Kbps" />
     <Parameter name="vmax" value="44928.0" label="$/month" />
   </LinearValuation>
  - <RobertsValuation current="false" label="Elastic Demand">
     <Parameter name="qmax" value="45000.0" label="Kbps" />
     <Parameter name="vmax" value="4928.0" label="$/month" />
   </RobertsValuation>
  - <BudgetValuation current="true" label="Budget Valuation">
     <Parameter name="qmax" value="1000.0" label="Kbps" />
     <Parameter name="budget" value="100.0" label="$/month" />
   </BudgetValuation>
```

```
- <WebValuation label="Web Valuation">
    <param name="delay" value="100.0" />
    <param name="hitspermonth" value="100000.0" />
    <param name="filesize" value="1000.0" />
    <param name="centsperhit" value="0.1" />
    <param name="randomize" value="false" />
  </WebValuation>
  <Parameter name="budget" value="51840.6" label="$/month" />
  <ManualStrategy current="false" label="Manual" />
  <TruthfulStrategy current="true" label="Auto" />
  <resourceAgentURL nickname="RESOURCE_NAME"
    current="true">http://HOSTNAME:HTTP_PORT/bx/RESOURCE_NAME</resourceAge
  <uploadURL nickname="HOSTNAME
    garage">http://HOSTNAME:HTTP_PORT/bx/garage</uploadURL>
  <param name="playerInterval" value="2000" />
  <param name="timeout" value="2000" />
  <param name="timelabel" value="min" />
  <param name="currencylabel" value="c" />
  <param name="quantitylabel" value="Mbps" />
  <param name="debug" value="false" />
</AuctionPlayer>
```

Figure 11(b)

```
/*
 * File:            Truthful.java
 *
 * Remark:          Strategy for player with diminishing returns
 *
 * $Id: Truthful.java,v 1.16          07:43:19 cobe Exp $
 *
 *
*/

package ihn.merkato;
import org.w3c.dom.*;
import com.sun.xml.tree.XmlDocument ;

/**
 * The strategy that bids the truthful best reply as in Proposition 1 o
f
 * the PSP paper.
 * It will only submit the bid if utility will be increased by at least
 * epsilon.
 *<p>
 *@author Nemo Semret
 *(
 */
public class Truthful extends AuctionStrategy  {

  Bid tmp = createBid( ) ;
    /**
   * Finds truthful best reply as in Proposition 1 of
   * the PSP paper.
   * Sets the bid at the player if utility will be increased by at leas
t
   * epsilon.
   * <p>
   * If timelogging is enabled, this will write to the  player's
   * log a line with current time, bid, allocation, and utility,
   * at each call.
   * @see #epsilon
   * @see ihn.merkato.AuctionPlayer#setBid
   */

public boolean bid( )  {

     double lq=0, uq= getPlayer ( ) .getValuation ( ) .qmax ( ) , mq= (uq+lq) / 2,
       dq =    getPlayer ( ) .dq ( ) ;

     if (debug( ) )  {
        getPlayer ( ) .log ("q range =    ["+lq+
                            " , "+uq+"]  dq="+dq+
                            "  Q="+getPlayer ( ) .stuff ( ) ) ;
        getPlayer ( ) .addnews (" . " ) ;
     }
```

Example of Agent Strategy

Figure 12(a)

Truthful.txt

```
// see Proposition 1
int i=0 ;
double mp, dv;
while ( uq-lq >dq && i<20 ) {
  i++;
  mq = (lq=uq) / 2;
  /*
       if (mq < getPlayer ( ) .stuff ( ) -
       (getBidder ( ) .getBidList ( ) ) .demandAtPrice (
       getPlayer ( ) .dval (mq, mq+dq) ,
       getPlayer ( ) .getId ( ) ) )
       */
  // the following is equivalent and more general
  dv=getPlayer ( ) .getValutaion ( ) .dval (mq, mq+dq) ;
  mp=getBidder ( ) .getBidList ( ) .marketPrice (getPlayer ( ) .stuff ( )
                                                  -mq,
                                                  getPlayer ( ) .getId ( )
) ;
  if (debug ( ) )
    getPlayer ( ) .log ("i ="+i+" mq= "+mq+" dv="+dv+" mp="+mp ) ;

  if (dv>mp)
    lq=mq ;
  else
    uq=mq ;
}

tmp.bidderid = getPlayer ( ) .getId ( ) ;
tmp.price = Data.MAXPRICE;
tmp.qty = lq;

if (debug( ) )
  getPlayer ( ) .log ( " "+ i +" steps. q range = ["+lq
                       +","+uq+"] currentbid="+
                        getBidder ( ) .anteBid ( ) + " found "+tmp) ;

if (util (tmp) <0) {
    uq= tmp.qty ;
     lq=0;
}

i=0 ;
while (uq-lq>dq && i<20) {
  tmp.qty = (uq+lq) /2;
  i++ ;
  if (debug ( ) )
    getPlayer ( ) .log ("i="+i+" q=" + tmp.qty ) ;
  if (util (tmp) <0)
    uq= tmp.qty;
  else
    lq = tmp.qty ;
```

Example of Agent Strategy

Figure 12 (b)

```
}

// need this in case the above loop is just outside the budget
while (util ( tmp) <0 && tmp.qty>0 && i <40) {
   i++ ;
   tmp.qty -=dq;
   if (debug ( ) )
     getPlayer ( ) .log ("i="+i+" q="+tmp.qty) ;
}

if (debug ( ) )
     getPlayer ( ) .log (""+i+" steps. range=[ "+lq+" , "+uq+" ] currentbid="+
                                        getBidder ( ) .anteBid ( ) . qty ) ;
1232
tmp.price = getPlayer ( ) . getValuation ( ) .dval ( tmp.qty, tmp.qty+dq) ;

double u = getPlayer ( ) .currentUtil ( ) ;
double newu = util (tmp) ;

if (debug ( ) ) {
   getPlayer ( ) .log ("currentalloc="+
                              getBidder ( ) .currentAllocation ( ) +
                              " newbid="+tmp+" antebid="+
                              getBidder ( ) .antBid ( ) ) ;
   getPlayer ( ) .log ("u="+u+" newu="+newu+" ante="
                        +util (getBidder ( ) .anteBid ( ) )
                        +" fee="+getBidder ( ) .bidFee ( )
                        +" epsilon="+epsilon ( )
                        +" avgdur="+getAvgDuration ( ) ) ;
}

if (getPlayer ( ) .trace ( ) ) {
   Bid alloc = getBidder ( ) .currentAllocation ( ) ;

   getPlayer ( ) .log (" "+getBidder ( ) .anteBid ( ) .qty
                              +"\t"+getBidder ( ) .anteBid ( ) .price+"\t"+
                              alloc.qty+"\t"+alloc.price+"\t"+
                              getPlayer ( ) .currentUtil ( ) ) ;
}

if ( newu > u + epsilon ( ) ) {
     if (debug ( ) ) getPlayer ( ) .addnews ( " * " ) ;
1234
     return getBidder ( ) .setBid (tmp.qty, tmp.price) ;
}
else {
     if (debug ( ) ) getPlayer ( ) .addnews ( " - " ) ;
     return false ;
}
}
}
```

Example of Agent Strate

Figure 12 (c)

```
/*
 *  BidList object
 *
 * File:          PSPBidList.java
 *
 *
 *
 *
 *
 *
 *
 *
 */

package ihn.diffpex;

import  ihn.merkato.Bid;
import  ihn.merkato.Data;
/**
 *
 *
 */
public class PSPBidList extends ihn.merkato.GenericBidList  {

   /**
    * Compute an allocation given the current profile of opponents
in
    * this bidlist. This class uses the Progressive-Second-Price
    * auction rule.
    * @param tb The bid for which the allocation is to be calculate
d.
    * @param Q  The total quantity of resource available.
    * /

   public   Bid allocation (Bid tb,  double Q)  {
     return PSPallocation (tb,  Q) ;
   }

   /**
    * Cumpute an allocation given the current profile of opponents
in
    * this bidlist,  with the Progressive-Second-Price
    * auction rule.
    * @param tb The bid for which the allocation is to be calculate
```

Figure 3(a)

```
d.
   * @param Q  The total quantity of resource available.
   * /

  private  synchronized Bid PSPallocation (Bid tb, double Q)  {
    Bid index = top;
    Bid alloc= new Bid ( ) ;
    double leftover = Q;      / /leftover with player id
    double leftoverwo =Q;  / /leftover without player id
    double qAj , qAj0 , num=0, den=0 ;
    boolean gotcha = false ;

    alloc.qty = Math.min (tb.qty,
                          Math.max (Q-demandAtPrice (tb.price,  tb.bid
derid) , 0) ) ;

    while(index.next  != null)  {
       index = index.next;

       if(index.bidderid  != tb.bidderid)  {

         if (index.price <= tb.price && !gotcha)  {
           leftover -= tb.qty ;
           if (leftover <=0) leftover=0 ;
           gotcha = true ;
         }

         qAj  =  (index.qty <= leftover ? index.qty : leftover) ;
         qAj0=  (index.qty <=  leftoverwo ? index.qty : leftoverwo) ;
         num +=  (index.price*  (qAj0- qaj) ) ;
         / /          den  += ( qAj0- qAj) ;

         leftoverwo   - = index.qty;
         leftover - = index.qty ;
         if (leftover <=0) leftover=0 ;
         if (leftoverwo <=0) leftoverwo=0 ;

    }


  }

  / /    if (!gotcha)  alloc.qty = (tb.qty<= leftover ? tb.qty :
```

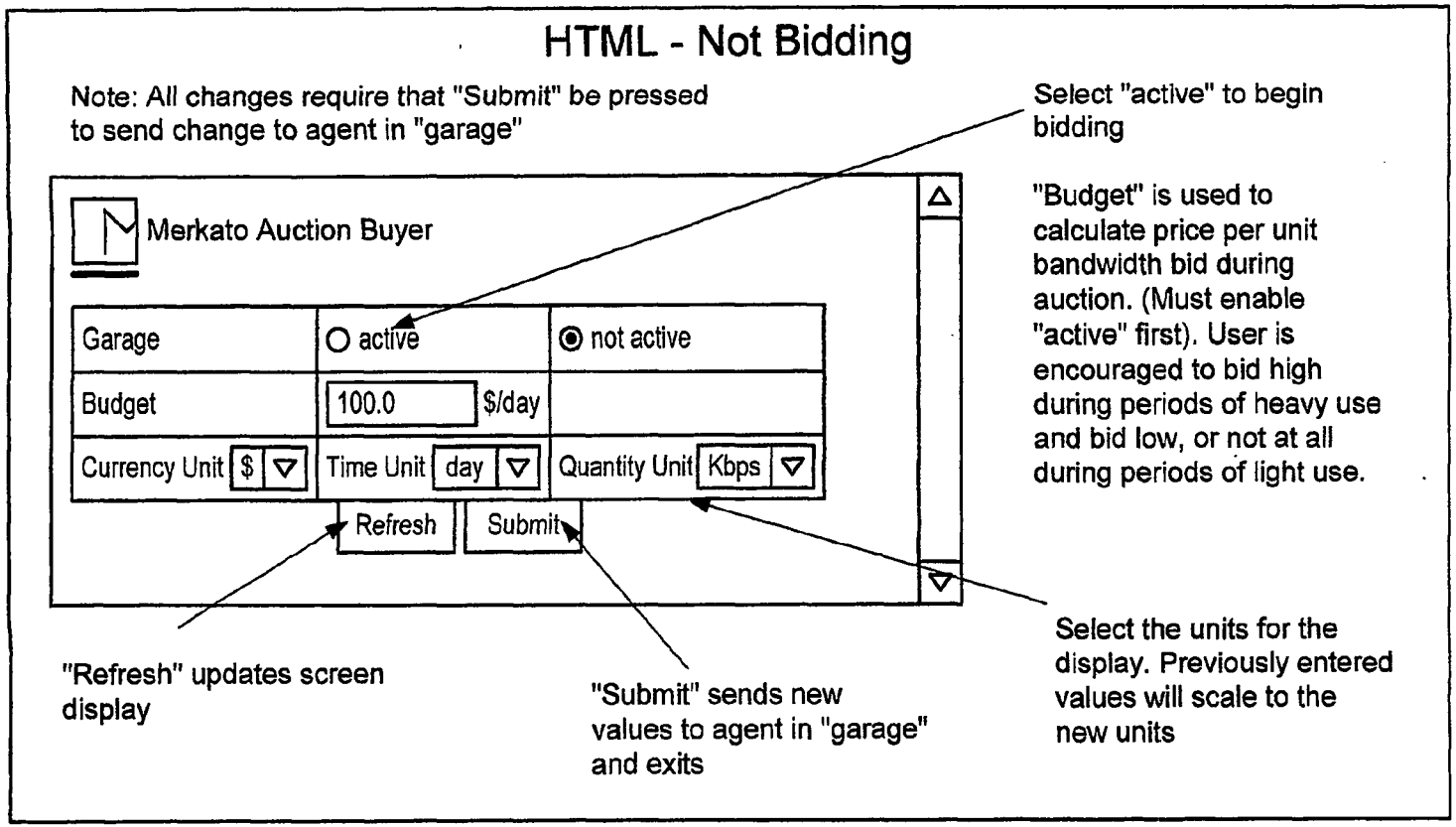
Figure 13(b)

```
leftover) ;

    //     alloc.price = den>0 ? num/den : 0 ;

    alloc.price = alloc.qty>0 ? num/alloc.qty : 0 ;
    alloc.bidderid = tb.bidderid ;

    return alloc;
}

    /**
     * Bids with ID#0 are not counted.
    */
public double revenue (double Q) {
    Bid index = top;
    double r=0 ;
    int I=0 ;
    Bid al ;
    while (index.next != null) {
        index = index.next;
        I + + ;
        if (index.bidderid!=0) {
            al = allocation (index, Q) ;
            r+= al.qty*al.price;
                // value(index.bidderid,Q) ;
        }
    }
    return r;
    //    if (I==0) return 0 ;
   //    else  return r -= (I-1) *value(Data.NOBODY, Q) ;
  }


}
// substituted 8 float to double
```

Figure 13(c)

```
<?xml version "1.0" encoding ="UTF-8" ?>
- <GenericAuctionAgent
   context="http://HOSTNAME:HTTP_PORT/bx/RESOURCE_NAME">
  <PlayerIdentity name="RESOURCE_USER" passwd="RESOURCE_PASSWD"
    ipaddress="127.0.0.1" netmask="255.255.255.255" />
 - <PSPBidList>
     <param name="randomduration" value="false" />
     <param name="duration" value="60000" />
     <param name="mustconv" value="true" />
     <param name="bidfee" value="0.01" />
     <param name="capacity" value"20000.0" />
  </PSPBidList>
  <UnixCryptAuthenticator passwdfile="MERKATO_HOME/accounts/passwd" />
 - <LinearValutaion>
     <parameter name="qmax" value="QMAX_VAL" label="QMAX_UNITS" />
     <parameter name="vmax" value="VMAX_VAL" label="VMAX_UNITS" />
  </LinearValuation>
  <param name="accountingDriverClass"
    value="ihn.merkato.AccountManager" />
  <param name="accountFile"
    value="http://HOSTNAME:HTTP_PORT/bx/dbstub" />
  <param name="hwDriverClass" value="RESOURCE_DRIVER_CLASS" />
  <param name="hwDevice" value="RESOURCE_DRIVER_INIT" />
  <param name="maxNBids" value="100" />
  <param name="verbose" value="true" />
  <param name="rememberIds" value="false" />
  <param name="clientTimeout" value="60000" />
  <param name="serverTimeout" value="30000" />
  <param name="pause" value="5000" />
  <param name="detailedlog" value="true" />
  <parameter name="maxBidFee" value="1.0" label="$" />
  <parameter name="maxAccountBalance" value="10000.0" label="$" />
</GenericAuctionAgent>
```

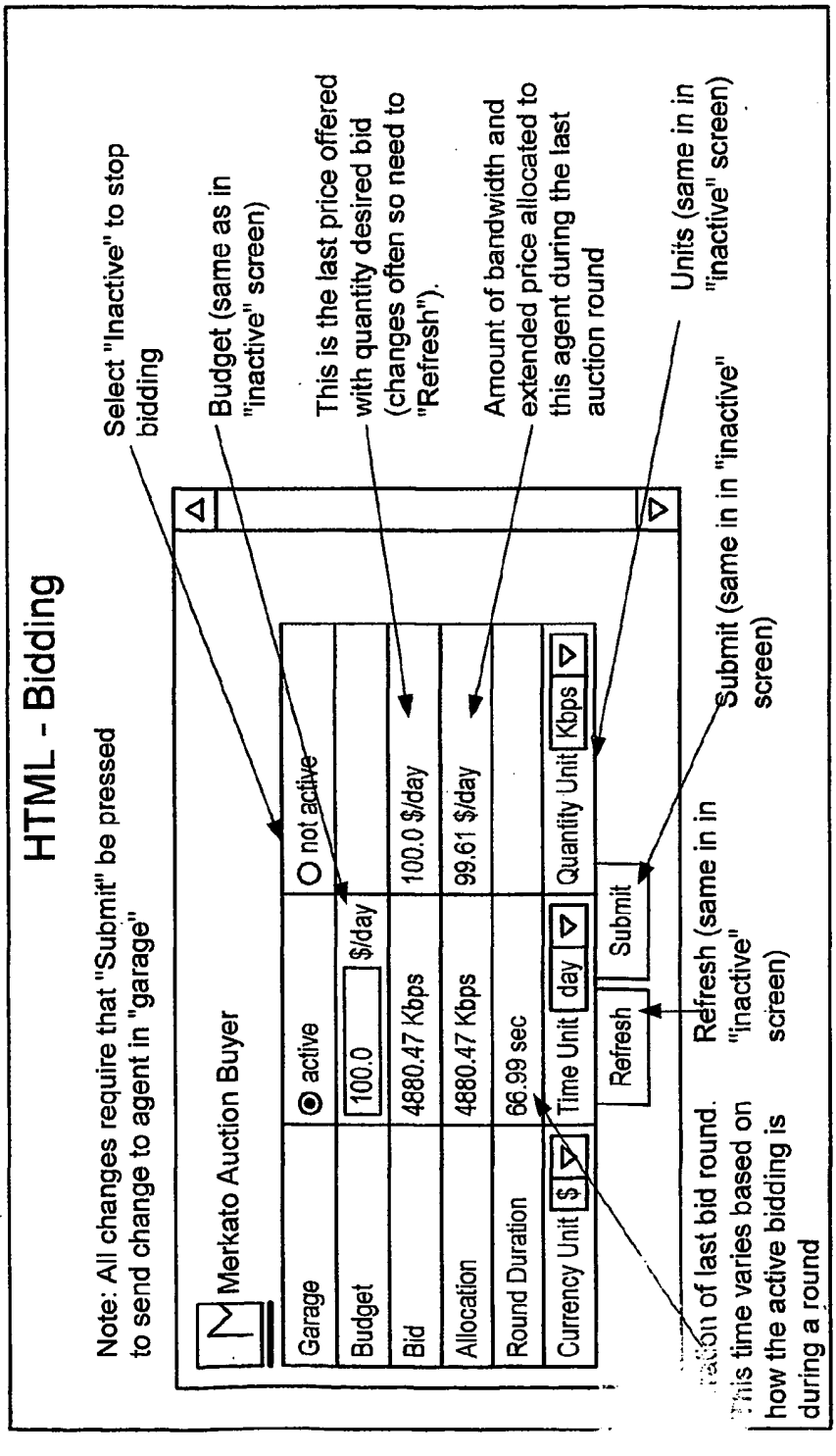
Figure 14



There is no cost accrued to buyers who are not bidding. They will be placed in the best-effort queue until they elect to bid for bandwidth.

... the budget value of the garaged agent to what you have entered into this screen and exits. At this point, it exits to a ... Merkato screen, but for customers, it will exit to a StreamingHand page.

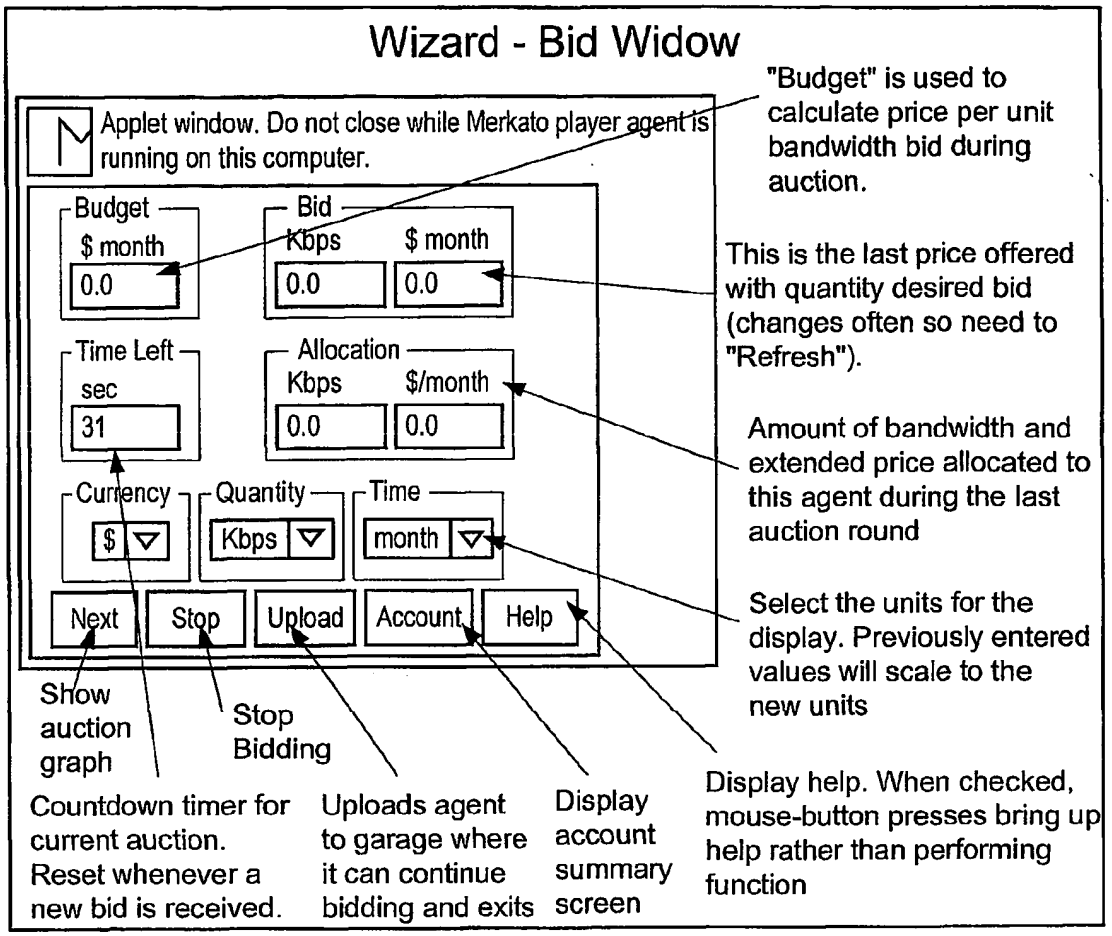
Figure 15(a)



The agent will attempt to obtain as much bandwidth as possible without exceeding budget. Conversely, the agent will request smaller and smaller amounts of bandwidth until they can obtain something for the budget price.

"Refresh" updates the screen. It does not send any changes to the "budget" value to the garaged agent. "Submit" does this (and then exits).

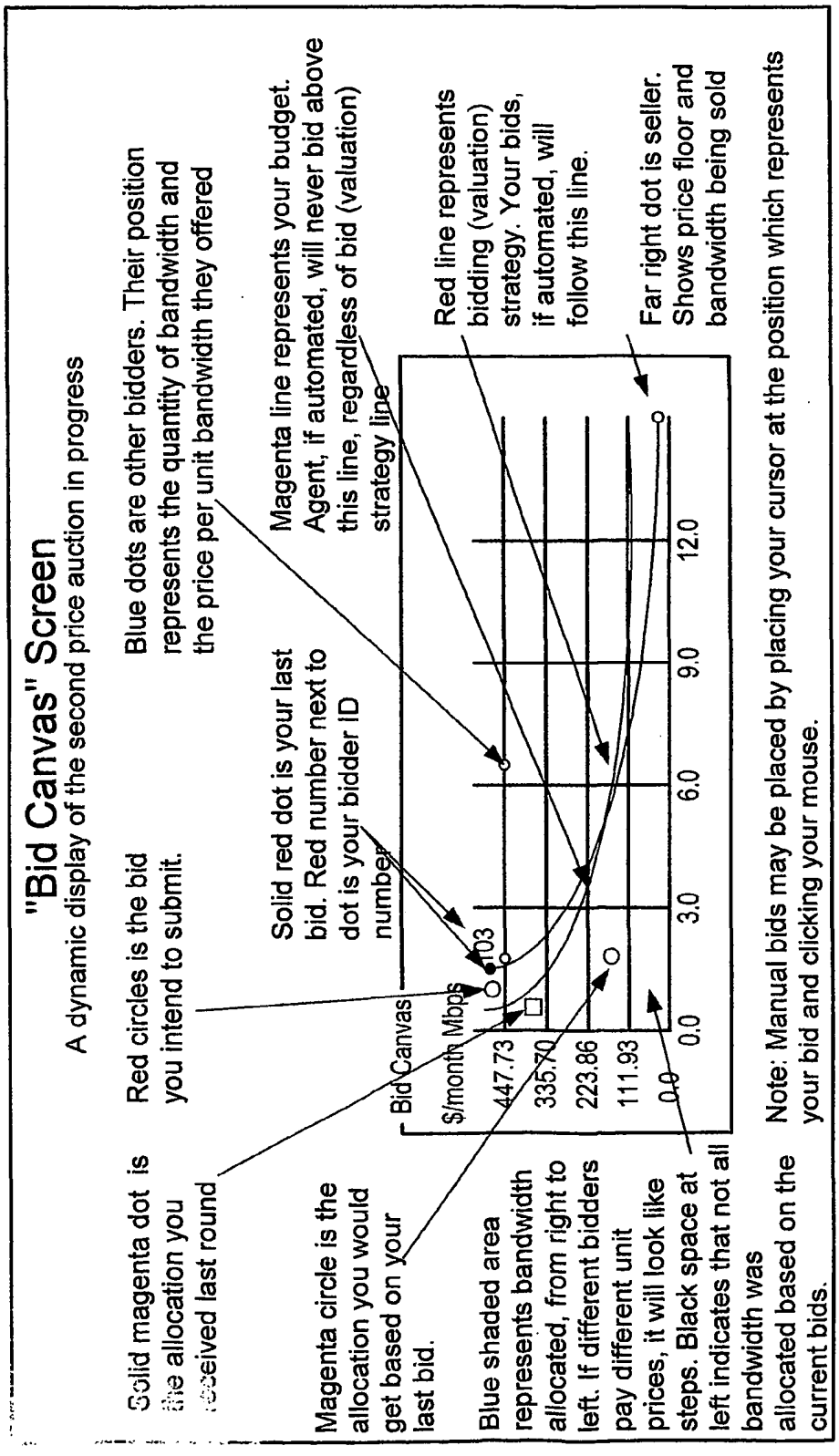
Figure 15(b)



"Stop" means to stop bidding. This bidding agent will not be charged and they will be placed in the shared best-effort queue.
"Upload" uploads the configuration to the garaged agent. Not that this will change some advanced settings to those assumed by this simple valuation and strategy model.
This simple budget-based valuation model has the bidding agent attempt to get as much bandwidth as possible without exceeding the budget number.
The strategy is based on the formula: price-per-unit-bandwidth * bandwidth-allocated=total-price-paid
Where the total-price-paid ("budget") is held constant, and the other two variables allowed to be altered.
Following this strategy, the bidder will first attempt to get all the bandwidth the seller is offering for their budgeted amount, which works out to the lowest possible price-per-unit-bandwidth. If unsuccessful, the bidding agent gradually increases the the offered price-per-unit-bandwidth and decreases the desired amount of bandwidth, until they successfully win an allocation.
If all bidders follow this valuation model, they will each get a bandwidth allocation that is the same proportion to total bandwidth as their budget is to the combined budgets of all bidders.

From the "Help" screen"
- Press **Start** to tell your agent to start bidding for you.
- Press **Stop** to tell your agent to stop

Figure 15(c)



Red Valuation Line and Magenta Budget Line are superimposed when "Budget" valuation is being used (default for "HTML" and "Wizard" Agent Interfaces).

Often the Red circle, red dot and magenta circle will be very close together.

Figure 15(d)

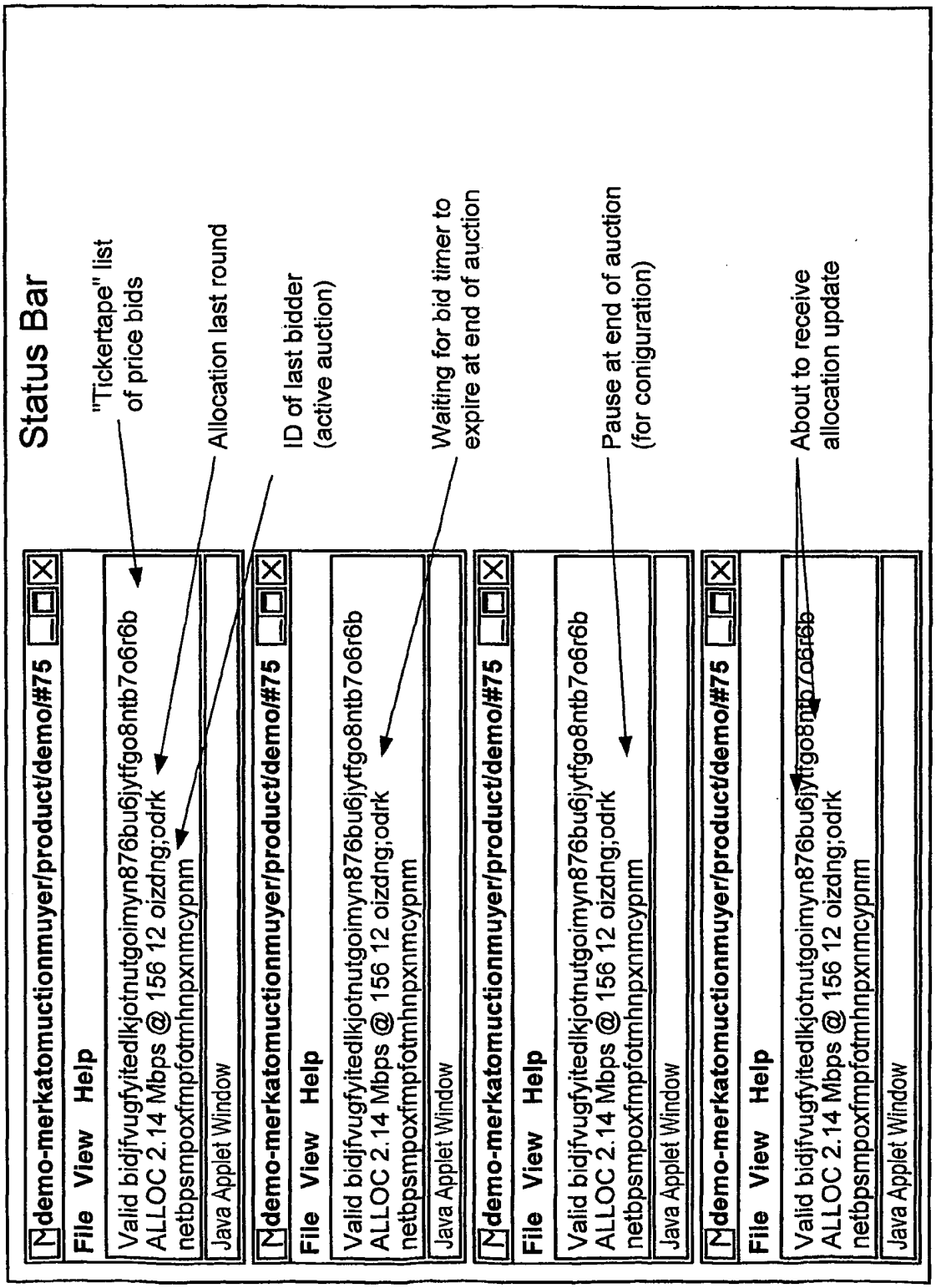

Status Bar
"Tickertape" list of price bids
Allocation last round
ID of last bidder (active auction)
Waiting for bid timer to expire at end of auction
Pause at end of auction (for coniguration)
About to receive allocation update
demo-merkatomuctionmuyer/product/demo/#75
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Java Applet Window

Figure 15(e)

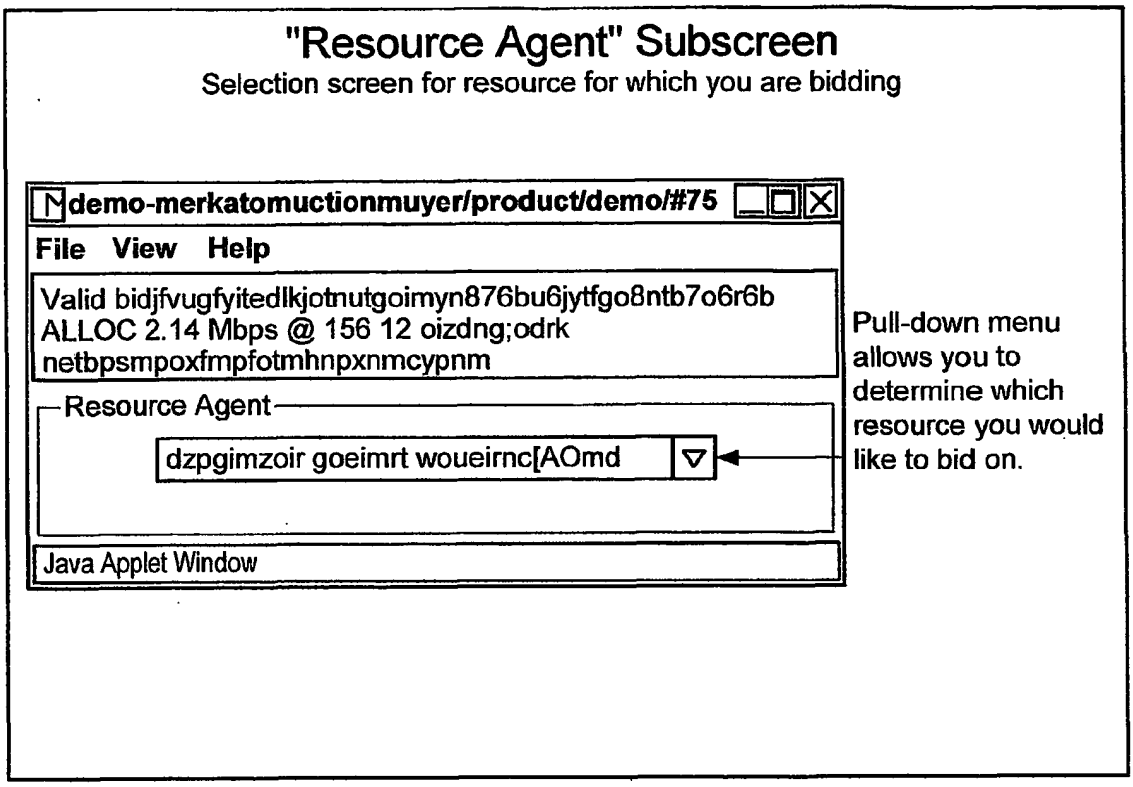

"Resource Agent" Subscreen
Selection screen for resource for which you are bidding
demo-merkatomuctionmuyer/product/demo/#75
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Resource Agent
dzpgimzoir goeimrt woueirnc[AOmd
Java Applet Window
Pull-down menu allows you to determine which resource you would like to bid on.

Figure 15(f)

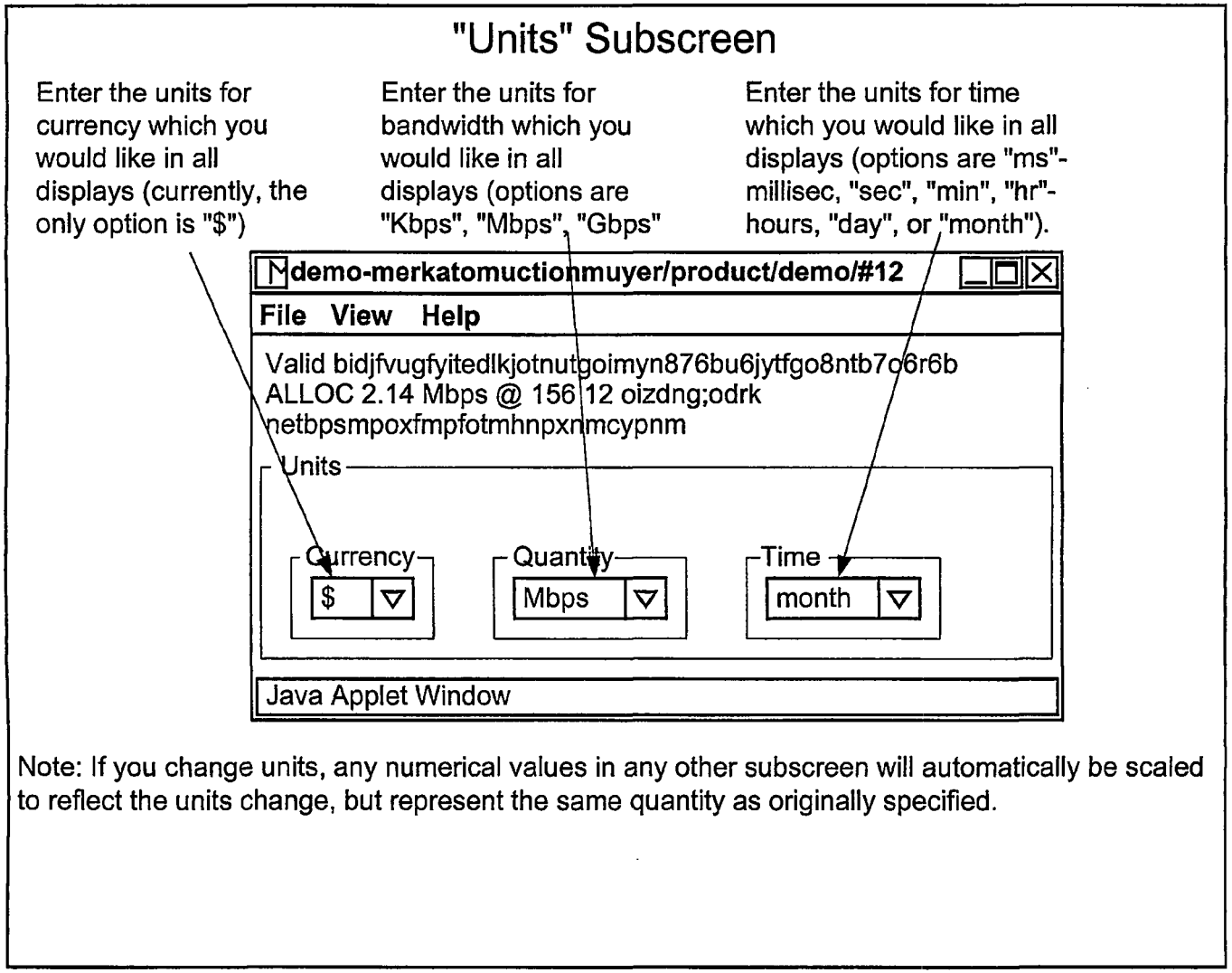

"Units" Subscreen
Enter the units for currency which you would like in all displays (currently, the only option is "$")
Enter the units for bandwidth which you would like in all displays (options are "Kbps", "Mbps", "Gbps"
Enter the units for time which you would like in all displays (options are "ms"-millisec, "sec", "min", "hr"-hours, "day", or "month").
demo-merkatomuctionmuyer/product/demo/#12
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Units
Currency
$
Quantity
Mbps
Time
month
Java Applet Window
Note: If you change units, any numerical values in any other subscreen will automatically be scaled to reflect the units change, but represent the same quantity as originally specified.

Figure 15(g)

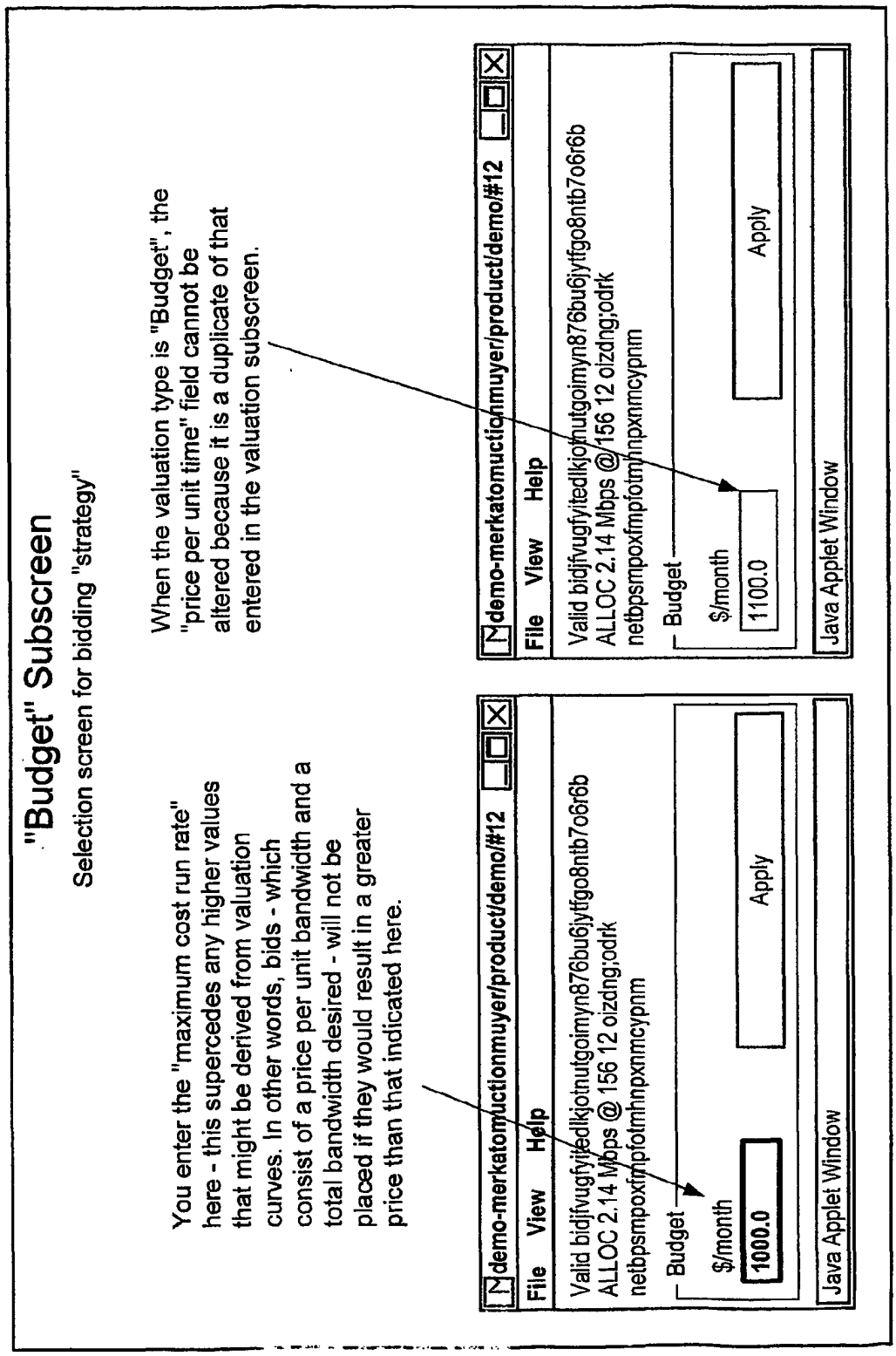

"Budget" Subscreen
Selection screen for bidding "strategy"
You enter the "maximum cost run rate" here - this supercedes any higher values that might be derived from valuation curves. In other words, bids - which consist of a price per unit bandwidth and a total bandwidth desired - will not be placed if they would result in a greater price than that indicated here.
When the valuation type is "Budget", the "price per unit time" field cannot be altered because it is a duplicate of that entered in the valuation subscreen.
demo-merkatomuctionmuyer/product/demo/#12
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Budget
$/month
1000.0
Apply
Java Applet Window
demo-merkatomuctionmuyer/product/demo/#12
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Budget
$/month
1100.0
Apply
Java Applet Window

Figure 15(h)

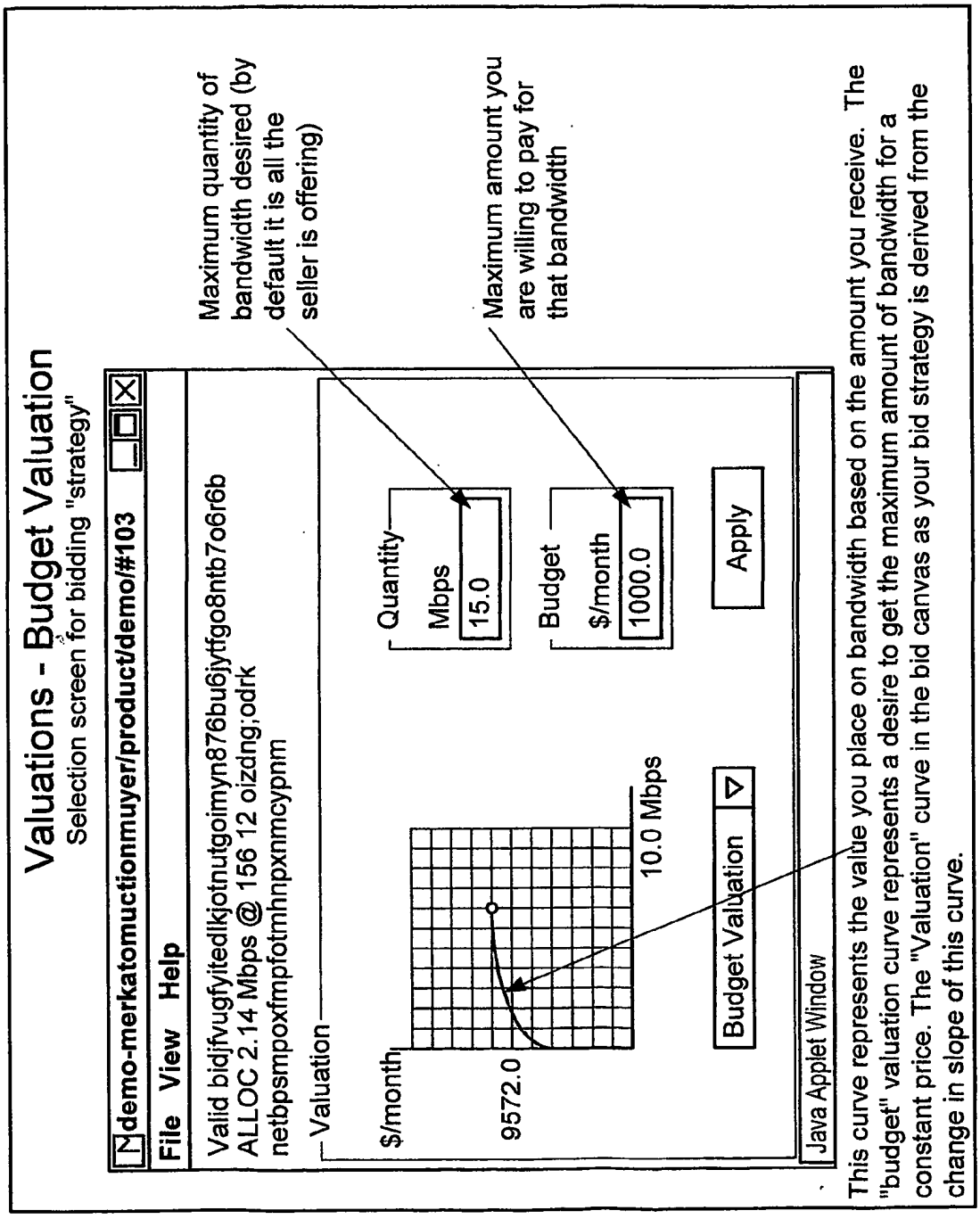

Valuations - Budget Valuation
Selection screen for bidding "strategy"
demo-merkatomuctionmuyer/product/demo/#103
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Valuation
$/month
9572.0
10.0 Mbps
Quantity
Mbps
15.0
Budget
$/month
1000.0
Apply
Budget Valuation
Java Applet Window
Maximum quantity of bandwidth desired (by default it is all the seller is offering)
Maximum amount you are willing to pay for that bandwidth
This curve represents the value you place on bandwidth based on the amount you receive. The "budget" valuation curve represents a desire to get the maximum amount of bandwidth for a constant price. The "Valuation" curve in the bid canvas as your bid strategy is derived from the change in slope of this curve.

Figure 15(i)

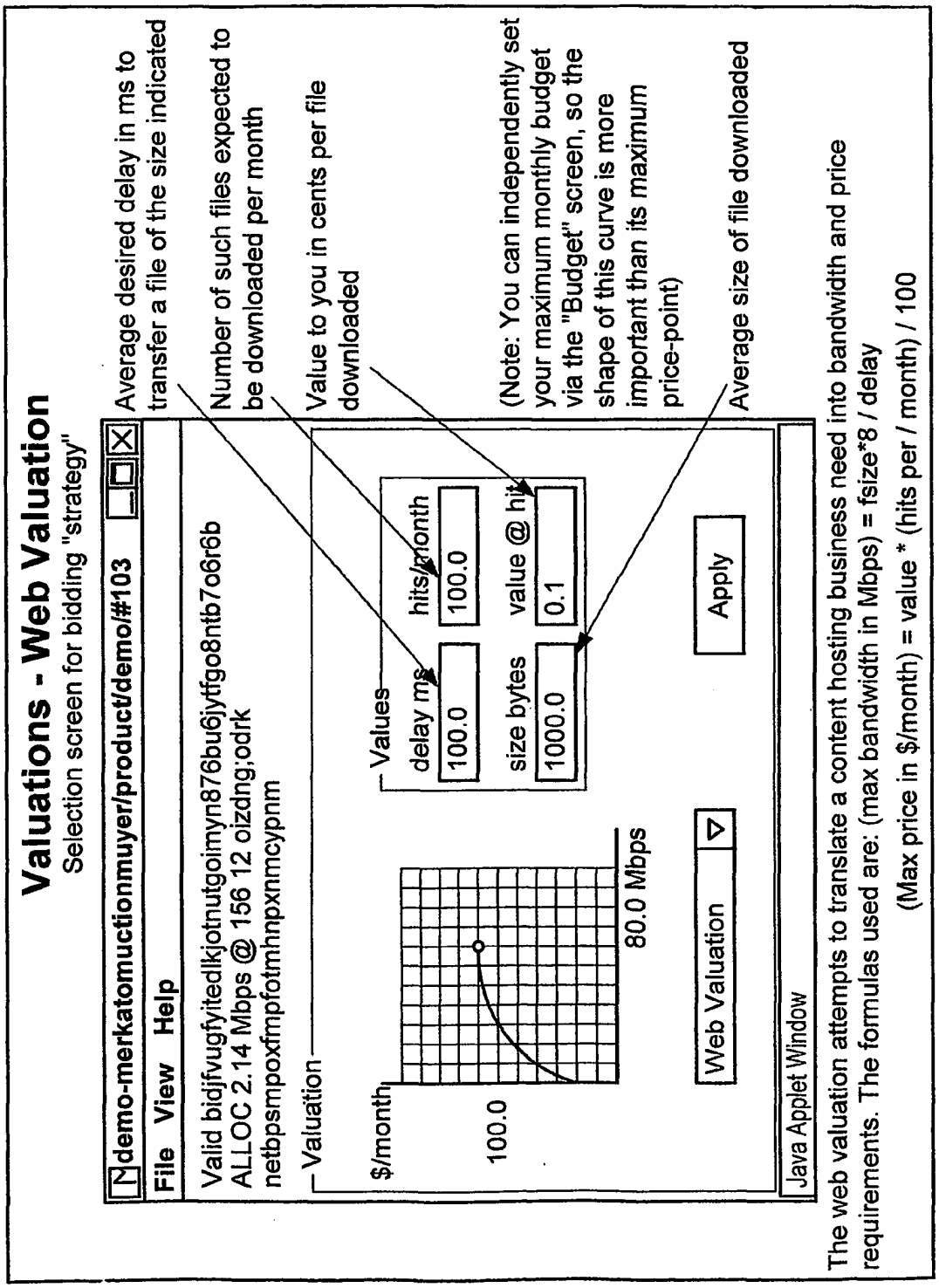

Valuations - Web Valuation
Selection screen for bidding "strategy"
demo-merkatomuctionmuyer/product/demo/#103
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Valuation
$/month
100.0
80.0 Mbps
Values
delay ms
100.0
hits/month
100.0
size bytes
1000.0
value @ hit
0.1
Web Valuation
Apply
Java Applet Window
Average desired delay in ms to transfer a file of the size indicated
Number of such files expected to be downloaded per month
Value to you in cents per file downloaded
(Note: You can independently set your maximum monthly budget via the "Budget" screen, so the shape of this curve is more important than its maximum price-point)
Average size of file downloaded
The web valuation attempts to translate a content hosting business need into bandwidth and price requirements. The formulas used are: (max bandwidth in Mbps) = fsize*8 / delay
(Max price in $/month) = value * (hits per / month) / 100

Figure 15(j)

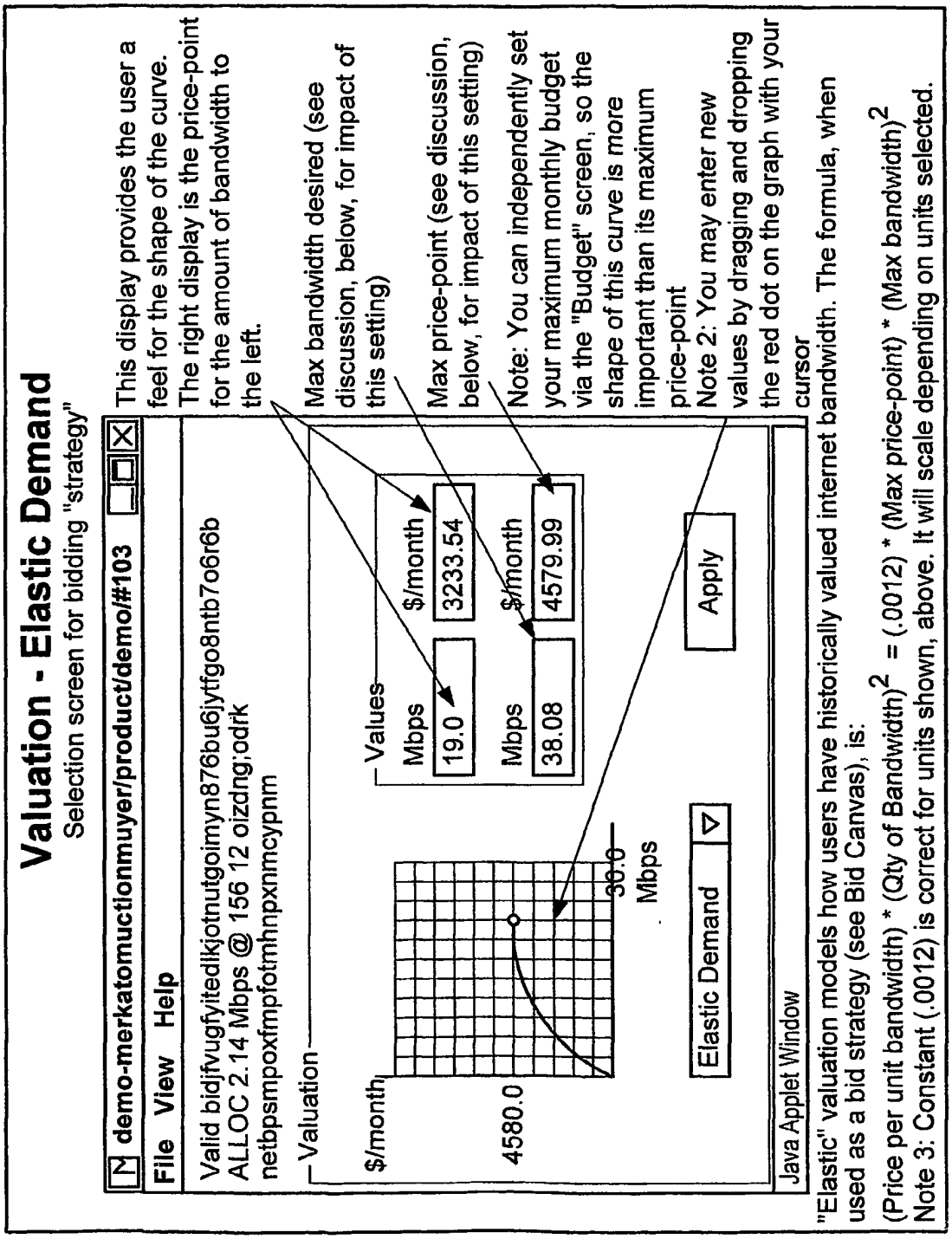
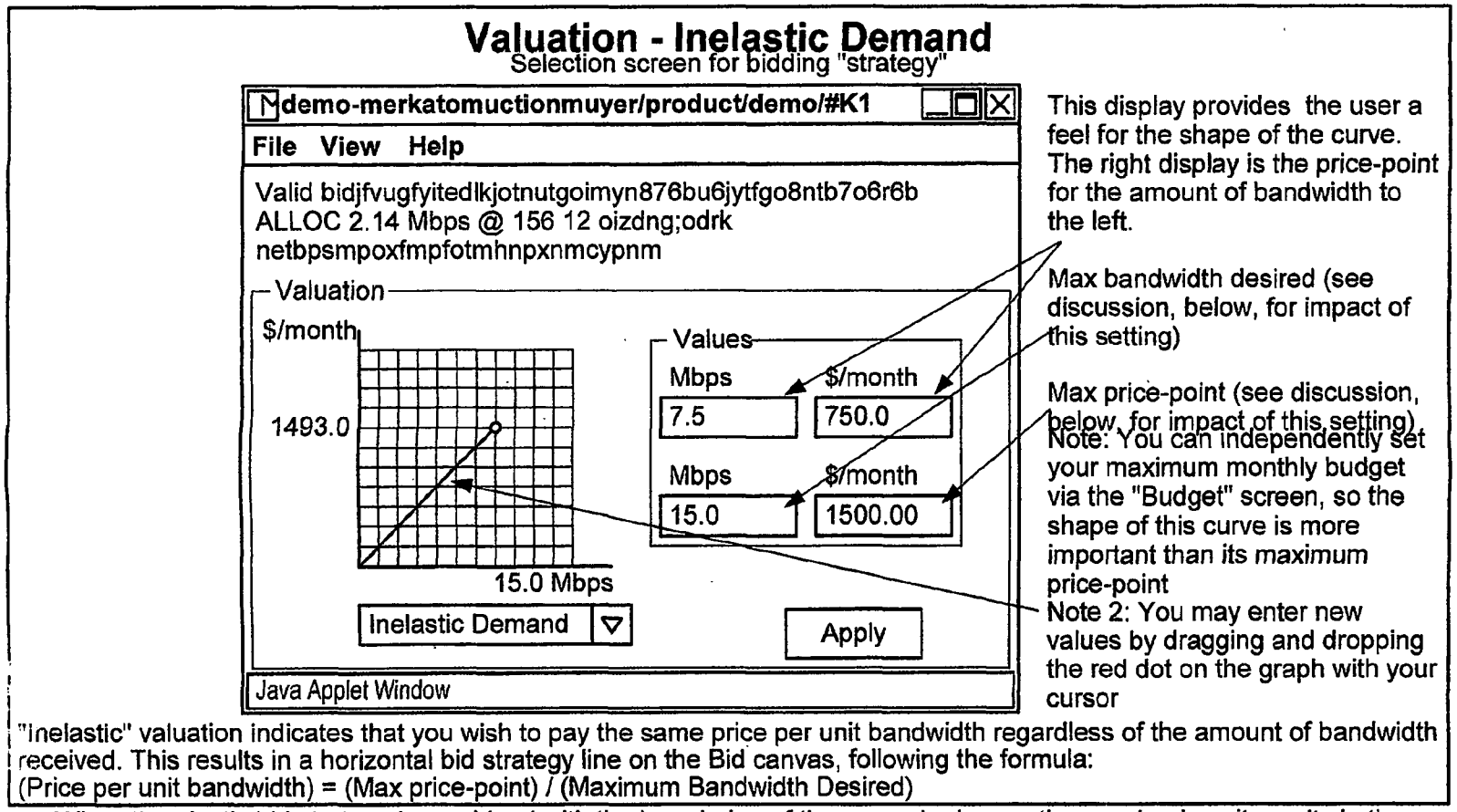

Valuation - Elastic Demand
Selection screen for bidding "strategy"
demo-merkatomuctionmuyer/product/demo/#103
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Valuation
$/month
4580.0
30.0
Mbps
Values
Mbps
19.0
$/month
3233.54
Mbps
38.08
$/month
4579.99
Elastic Demand
Apply
Java Applet Window
This display provides the user a feel for the shape of the curve. The right display is the price-point for the amount of bandwidth to the left.
Max bandwidth desired (see discussion, below, for impact of this setting)
Max price-point (see discussion, below, for impact of this setting)
Note: You can independently set your maximum monthly budget via the "Budget" screen, so the shape of this curve is more important than its maximum price-point
Note 2: You may enter new values by dragging and dropping the red dot on the graph with your cursor
"Elastic" valuation models how users have historically valued internet bandwidth. The formula, when used as a bid strategy (see Bid Canvas), is:
(Price per unit bandwidth) * (Qty of Bandwidth)^2 = (.0012) * (Max price-point) * (Max bandwidth)^2
Note 3: Constant (.0012) is correct for units shown, above. It will scale depending on units selected.

Figure 15(k)



When the elastic bid strategy is combined with the knowledge of the second price auction mechanism, it results in the following behavior:

If the elastic valuation is above the budget line, the agent will do a reverse calculation to determine when it can bid on the valuation line, but obtain the bandwidth on the budget line.

If the elastic valuation is below the budget line, the agent will continue to ask for the maximum amount of bandwidth at the valuation price and not accept a lesser amount of bandwidth.

Figure 15(l)

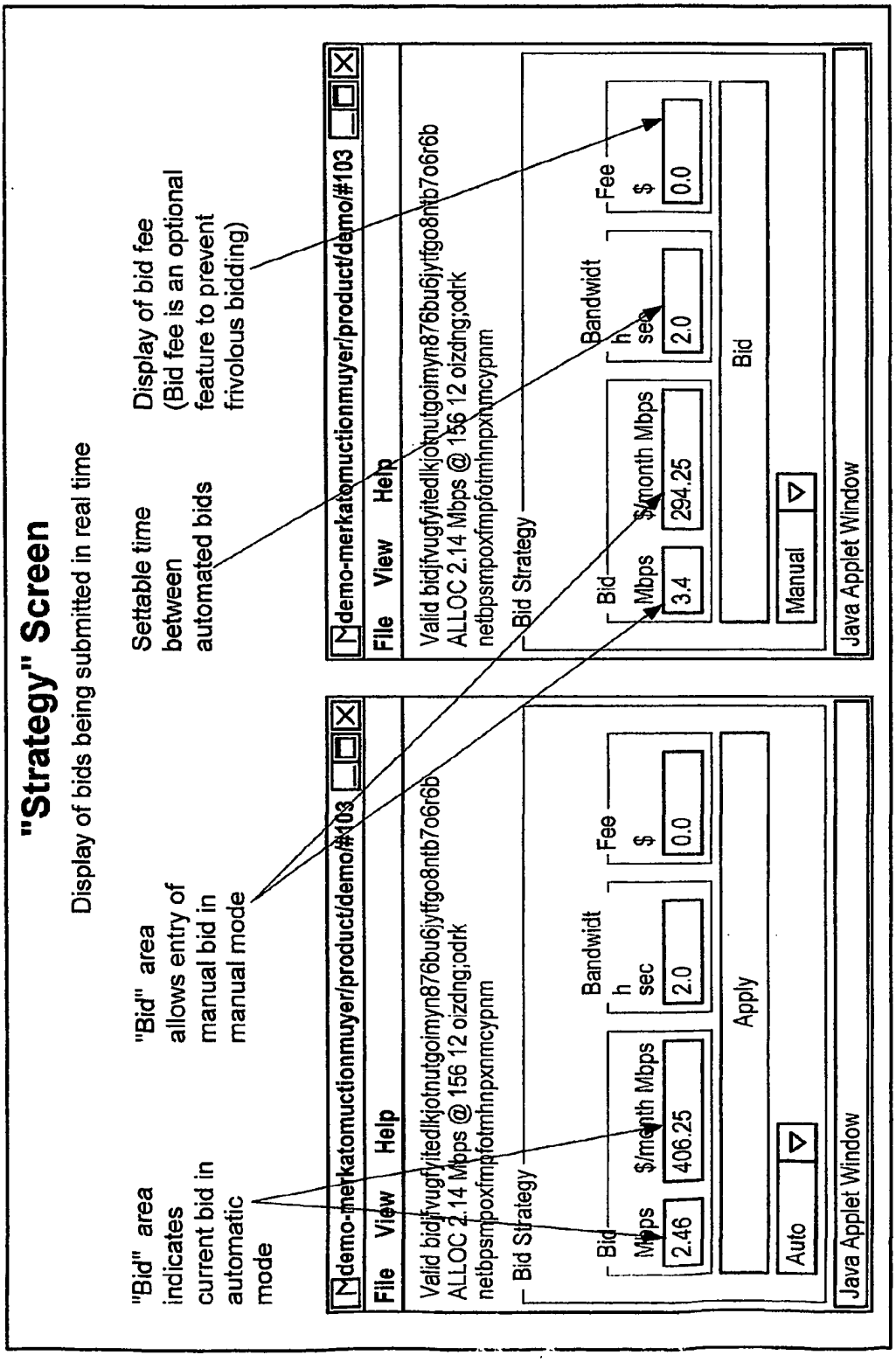

"Strategy" Screen
Display of bids being submitted in real time
"Bid" area indicates current bid in automatic mode
"Bid" area allows entry of manual bid in manual mode
Settable time between automated bids
Display of bid fee (Bid fee is an optional feature to prevent frivolous bidding)
demo-merkatomuctionmuyer/product/demo/#103
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jyftgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Bid Strategy
Bid
Mbps
$/month Mbps
2.46
406.25
Bandwidt
h
sec
2.0
Fee
$
0.0
Apply
Auto
Manual
Java Applet Window

Figure 15(m)

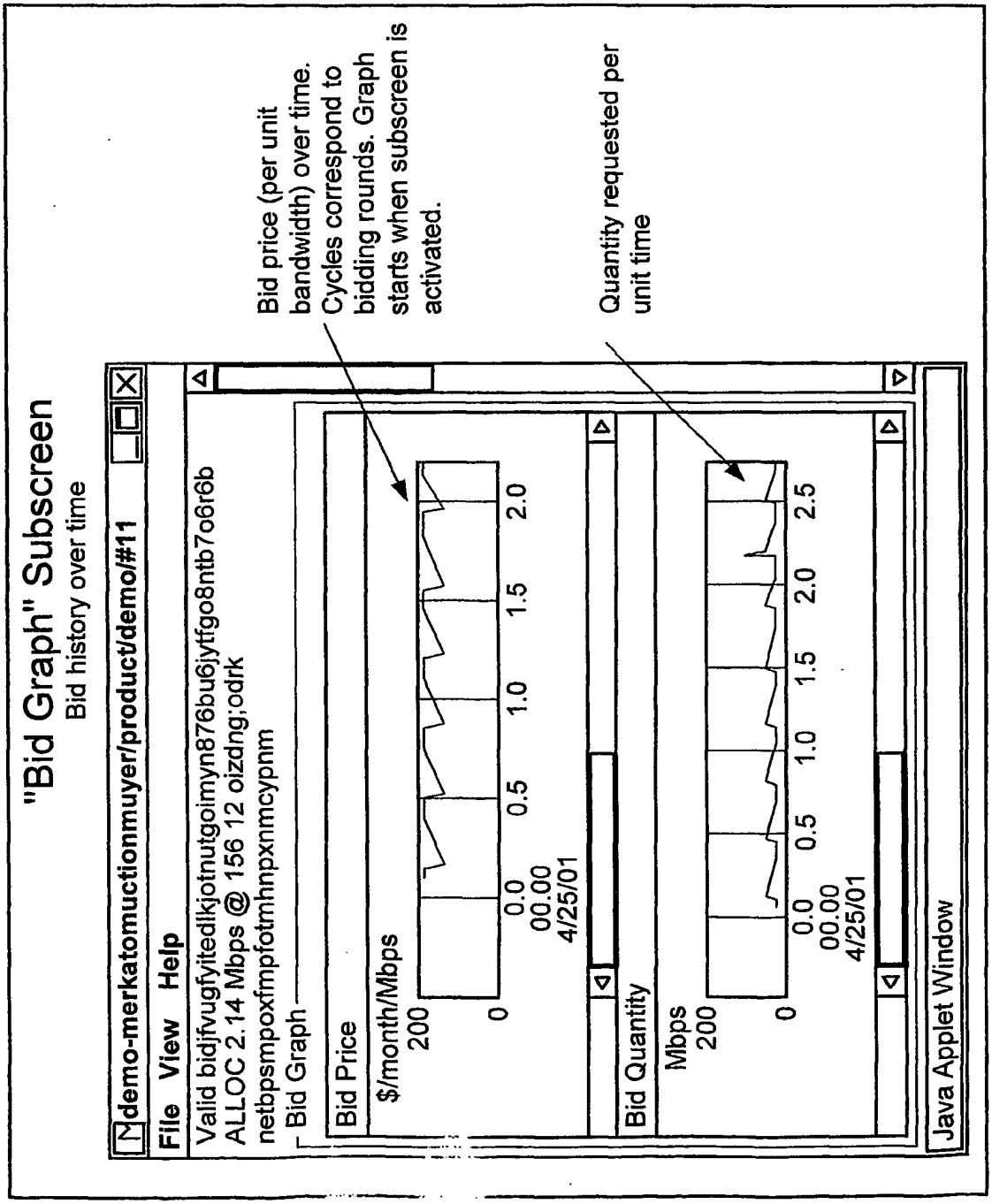

"Bid Graph" Subscreen
Bid history over time
demo-merkatomuctionmuyer/product/demo/#11
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Bid Graph
Bid Price
$/month/Mbps
200
0
0.0
00.00
4/25/01
0.5
1.0
1.5
2.0
Bid price (per unit bandwidth) over time. Cycles correspond to bidding rounds. Graph starts when subscreen is activated.
Bid Quantity
Mbps
200
0
0.0
00.00
4/25/01
0.5
1.0
1.5
2.0
2.5
Quantity requested per unit time
Java Applet Window

Figure 15(n)

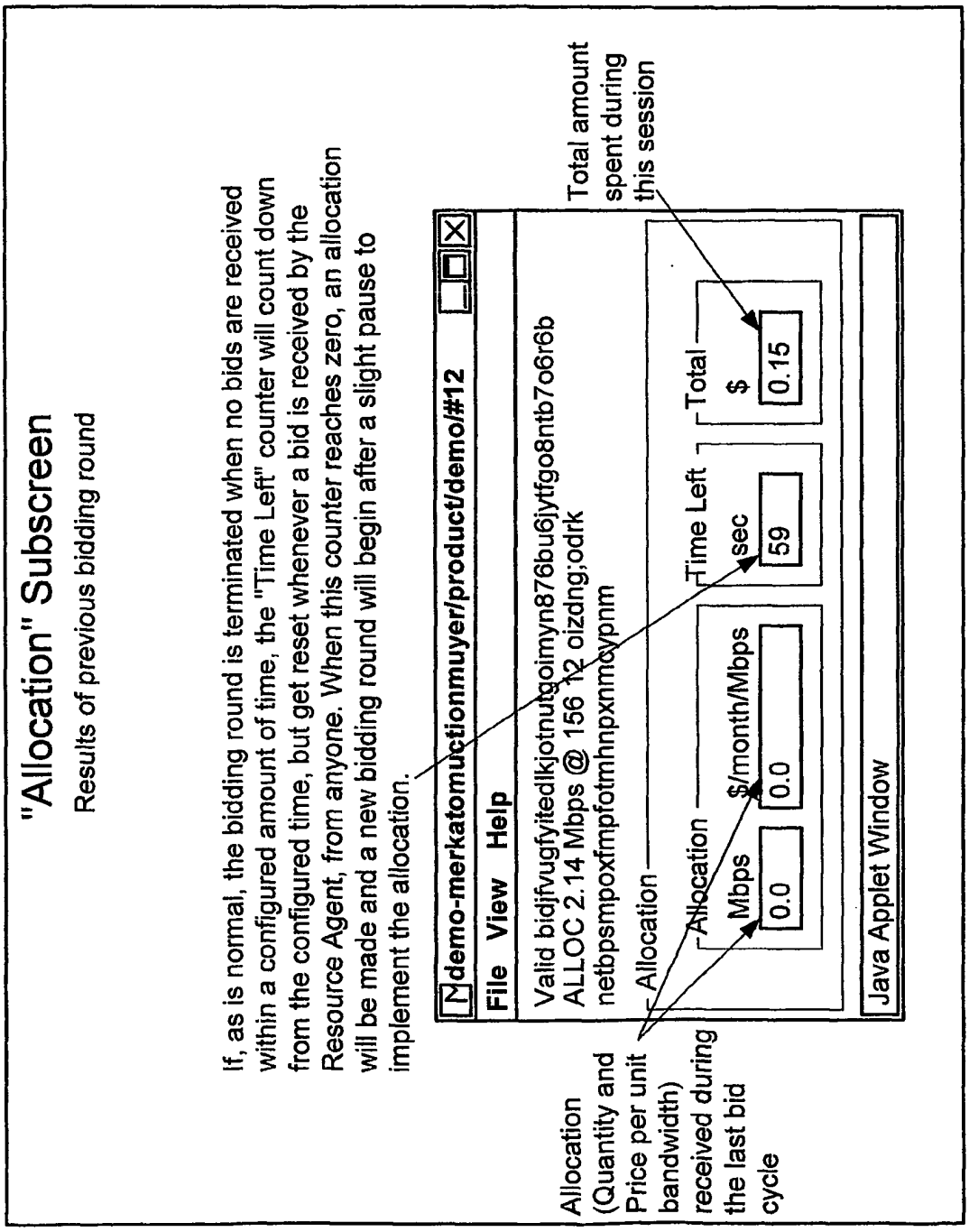

"Allocation" Subscreen
Results of previous bidding round
If, as is normal, the bidding round is terminated when no bids are received within a configured amount of time, the "Time Left" counter will count down from the configured time, but get reset whenever a bid is received by the Resource Agent, from anyone. When this counter reaches zero, an allocation will be made and a new bidding round will begin after a slight pause to implement the allocation.
demo-merkatomuctionmuyer/product/demo/#12
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmoypnm
Allocation
Allocation
Mbps
$/month/Mbps
0.0
0.0
Time Left
sec
59
Total
$
0.15
Java Applet Window
Allocation (Quantity and Price per unit bandwidth) received during the last bid cycle
Total amount spent during this session

Figure 15(o)

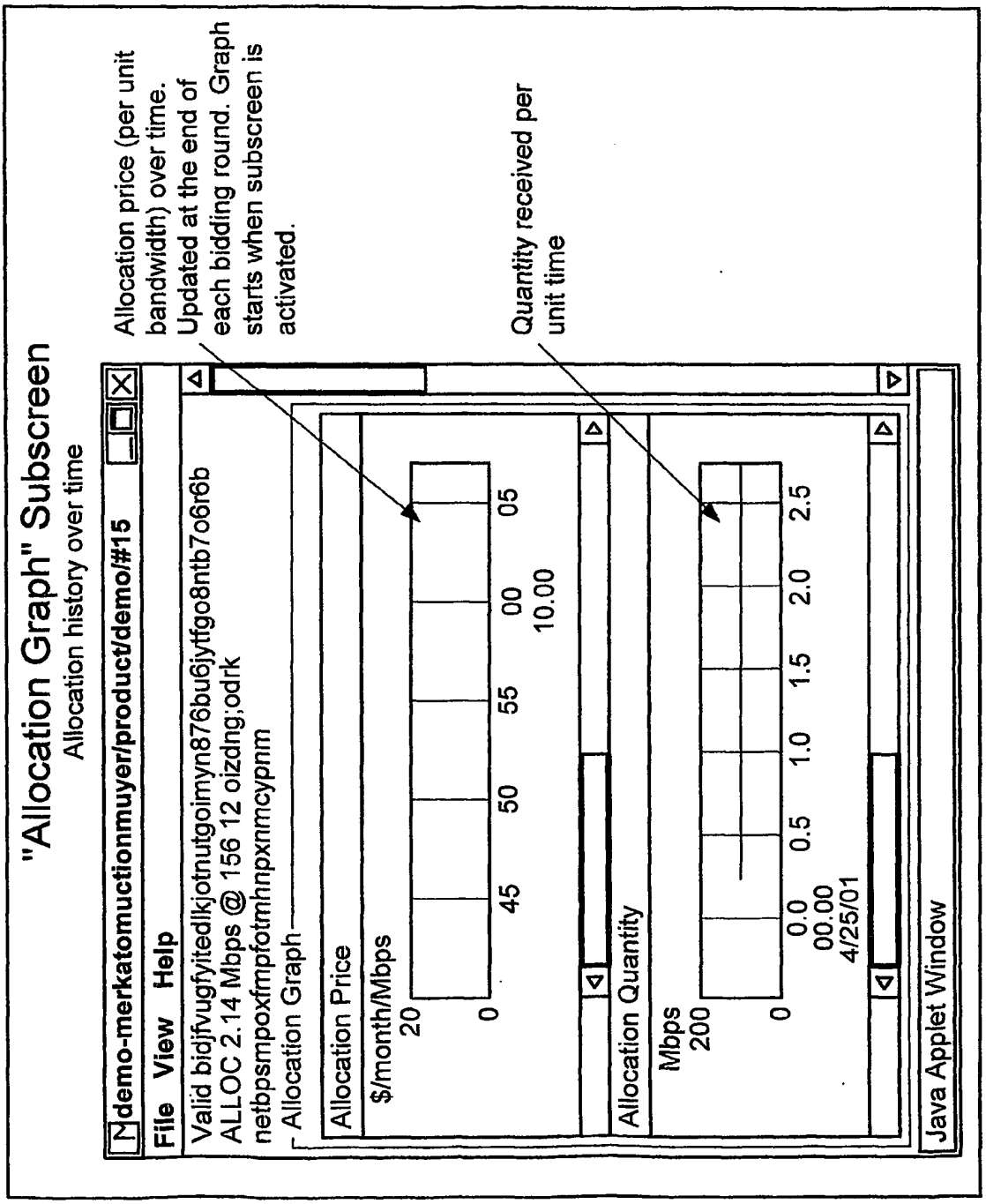

"Allocation Graph" Subscreen
Allocation history over time
demo-merkatomuctionmuyer/product/demo/#15
File View Help
Valid bidjfvugfyitedlkjotnutgoimyn876bu6jytfgo8ntb7o6r6b
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsmpoxfmpfotmhnpxnmcypnm
Allocation Graph
Allocation price (per unit bandwidth) over time. Updated at the end of each bidding round. Graph starts when subscreen is activated.
Quantity received per unit time
Allocation Price
$/month/Mbps
20
0
45
50
55
00
05
10.00
Allocation Quantity
Mbps
200
0
0.0
0.5
1.0
1.5
2.0
2.5
00.00
4/25/01
Java Applet Window

Figure 15(p)

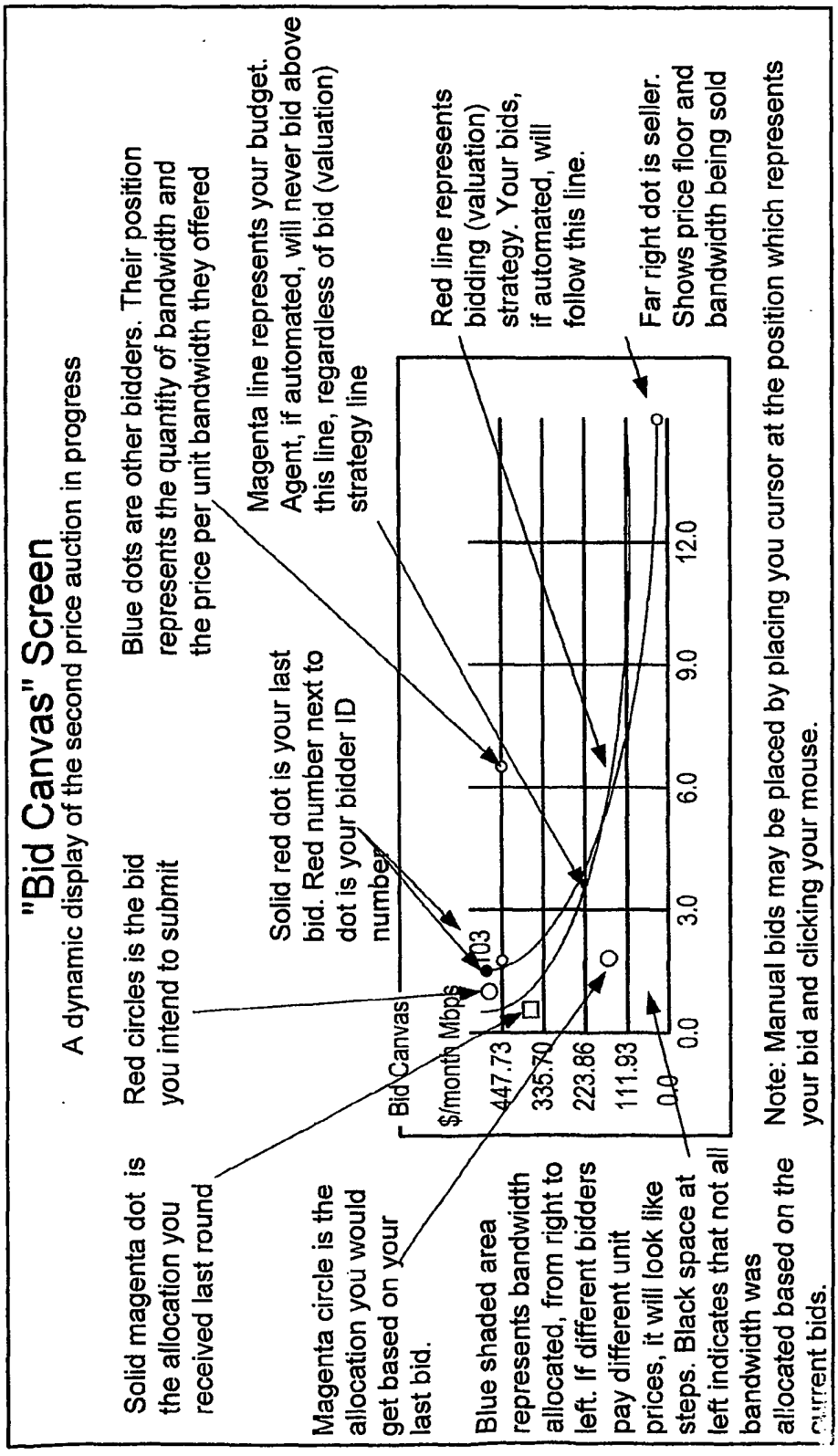

"Bid Canvas" Screen
A dynamic display of the second price auction in progress
Red circles is the bid you intend to submit
Blue dots are other bidders. Their position represents the quantity of bandwidth and the price per unit bandwidth they offered
Magenta line represents your budget. Agent, if automated, will never bid above this line, regardless of bid (valuation) strategy line
Solid red dot is your last bid. Red number next to dot is your bidder ID number
Red line represents bidding (valuation) strategy. Your bids, if automated, will follow this line.
Far right dot is seller. Shows price floor and bandwidth being sold
Solid magenta dot is the allocation you received last round
Magenta circle is the allocation you would get based on your last bid.
Blue shaded area represents bandwidth allocated, from right to left. If different bidders pay different unit prices, it will look like steps. Black space at left indicates that not all bandwidth was allocated based on the current bids.
Note: Manual bids may be placed by placing you cursor at the position which represents your bid and clicking your mouse.
Bid Canvas
$/month Mbps
447.73
335.70
223.86
111.93
0.0
0.0
3.0
6.0
9.0
12.0
03

Figure 15(q)

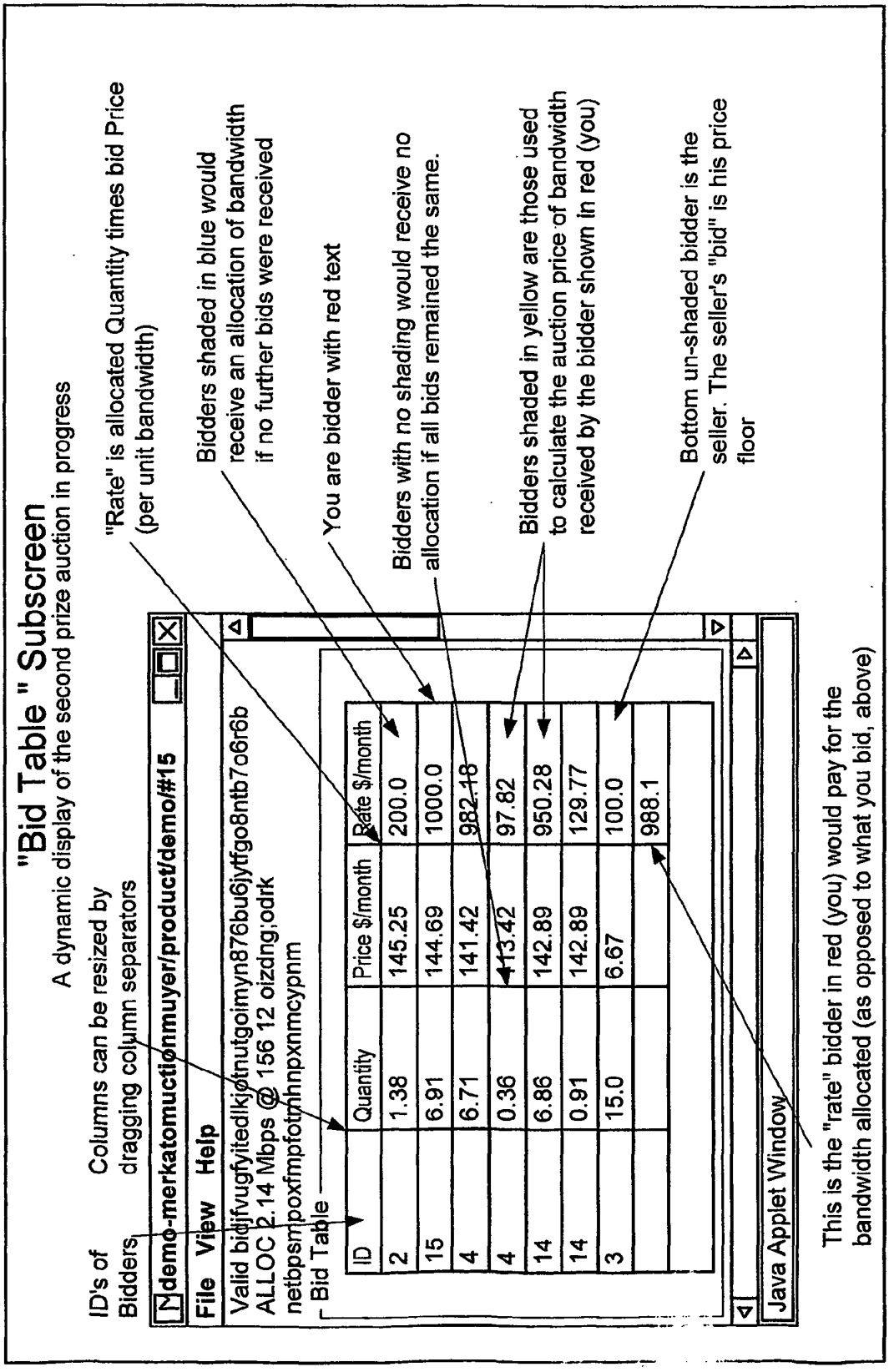

"Bid Table " Subscreen
A dynamic display of the second prize auction in progress
"Rate" is allocated Quantity times bid Price (per unit bandwidth)
Bidders shaded in blue would receive an allocation of bandwidth if no further bids were received
You are bidder with red text
Bidders with no shading would receive no allocation if all bids remained the same.
Bidders shaded in yellow are those used to calculate the auction price of bandwidth received by the bidder shown in red (you)
Bottom un-shaded bidder is the seller. The seller's "bid" is his price floor
ID's of Bidders
Columns can be resized by dragging column separators
demo-merkatomuctionmuyer/product/demo/#15
File View Help
Valid bid
ALLOC 2.14 Mbps @ 156 12 oizdng;odrk
netbpsm poxfmpfotmhnpxnmcypnm
Bid Table
ID
Quantity
Price $/month
Rate $/month
2 1.38 145.25 200.0
15 6.91 144.69 1000.0
4 6.71 141.42 982.48
4 0.36 113.42 97.82
14 6.86 142.89 950.28
14 0.91 142.89 129.77
3 15.0 6.67 100.0
988.1
Java Applet Window
This is the "rate" bidder in red (you) would pay for the bandwidth allocated (as opposed to what you bid, above)

Figure 15(r)